# Atlas

## Cross-Platform C++ Bitcoin Wallet

### Philip Glazman

#### Spring 2018

# Table of Contents

1. Introduction
2. Installation Instructions
3. User Manual
4. Design
5. Insights and Challenges
6. Conclusion
7. Bibliography
8. Source Code
9. Test Cases

# Introduction

##### Glossary

* Bitcoin – Network protocol used to reach consensus on who owns bitcoins.
* bitcoin – The value transferred in the Bitcoin protocol.
* Satoshis – The lowest denominator of bitcoin. One satoshi is 1/100millionth of a bitcoin.
* BIP – Bitcoin Improvement Proposals (BIP) are approved or pending proposals to the Bitcoin protocol. Several BIPs provide a standard for how the protocol or nodes should behave. This project uses several BIP standards regarding how wallets ought to be implemented.
* Mining – The process by which the network reaches consensus and a transaction is confirmed.
* Script – The programming language used by Bitcoin to write scripts. This language operates uses operation codes on a reverse polish notation stack.
* Smart Contract – A piece of code that is self-enforcing on the blockchain.
* Blockchain – Public data structure that maintains a ledger representing the entire state of the network.

For more information on these terms and others not covered, please consider reviewing the open-source Bitcoin wiki located here: <https://en.bitcoin.it/>

##### Background

Bitcoin is a digital money developed in 2009 where each node participating in the network can independently validate transactions and propagate them throughout the network using software similar to bittorrent. The protocol relies on public-key cryptography to create public addresses for the end-user. In terms of bitcoin, a wallet software manages the private keys that are associated with each public address. These keys gives users ownership in spending transactions and bitcoin. In bitcoin, the wallet is an abstraction that allows the end-user to send and receive payments.

##### Project

The aim of this project is to create a user-friendly bitcoin wallet implementation that encourages self-ownership of bitcoins and the use of bitcoin’s Script language. Several Bitcoin wallets exist in the ecosystem but there does not exist a wallet that provides an abstraction layer that allows the end-user to interact with bitcoin smart contracts in a user-friendly way.

Atlas proposes a different way for the user to interact with how interact with Bitcoin. With a focus on financial independence through education, Atlas provides a straightforward way for the user to write smart contracts and learn more about the underlying low-level protocol.

##### Warning

This wallet was not extensively tested for security vulnerabilities, therefore should not be used with real bitcoin. The current implementation of Atlas operates on the Bitcoin test network and uses test network bitcoins. Funds sent and received should be used with addresses that have a test network prefix.

##### Other Notes

This project heavily relied on Andreas M. Antonopoulos’s *Mastering Bitcoin*, open-source documentation notes on Libbitcoin on the Libbitcoin Wiki, and Aaron Jaramillo’s tutorials on Libbitcoin. These, among other scattered documentation along the web, were very helpful and resourceful. Several illustrations are used and referenced in this documentation that are from *Mastering Bitcoin*.

# Installation Instructions

Before running Atlas, a couple of important libraries are needed on the local machine.

1. Boost

Visit <https://www.boost.org/users/download/>

$ brew install boost

1. Libbitcoin

Visit <https://github.com/libbitcoin/libbitcoin/tree/version3>.

$ ./autogen.sh

$ ./configure

$ make

$ sudo make install

$ sudo ldconfig

More details on Libbitcoin installation can be found on Github README.md

1. Curl

Visit <https://curl.haxx.se/download.html>

$ brew install curl

1. JsonCPP

Visit <https://github.com/open-source-parsers/jsoncpp>

$ brew install jsoncpp

1. OpenSSL

Visit <https://www.openssl.org/>

$ brew install openssl

1. Run make file in /qt.

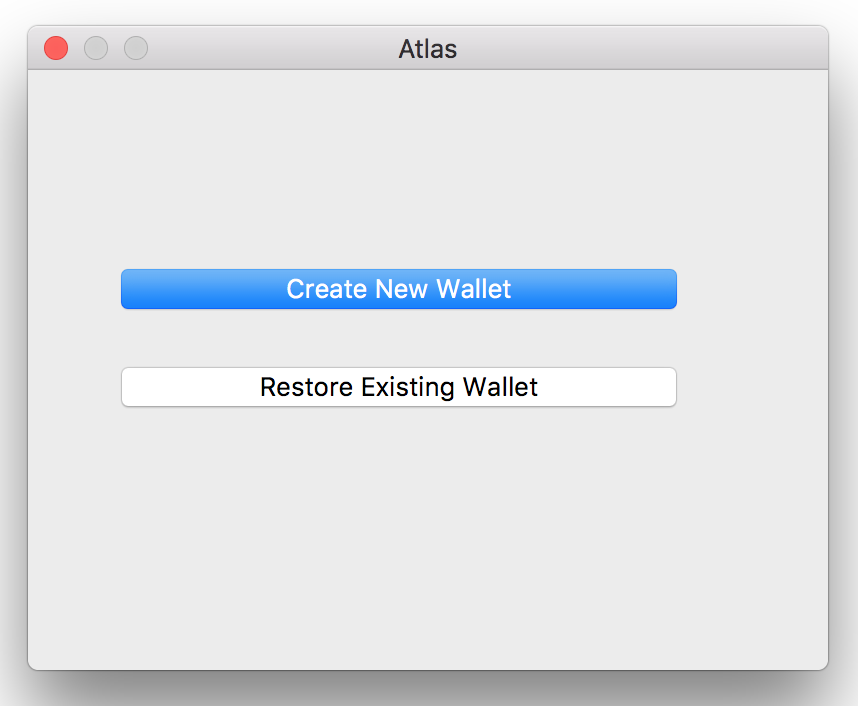
# User Manual

##### Warning

Atlas uses industry standards for wallet management but there are several risks involved. As noted in the warning section in the Introduction, this wallet should not be used with real bitcoin.

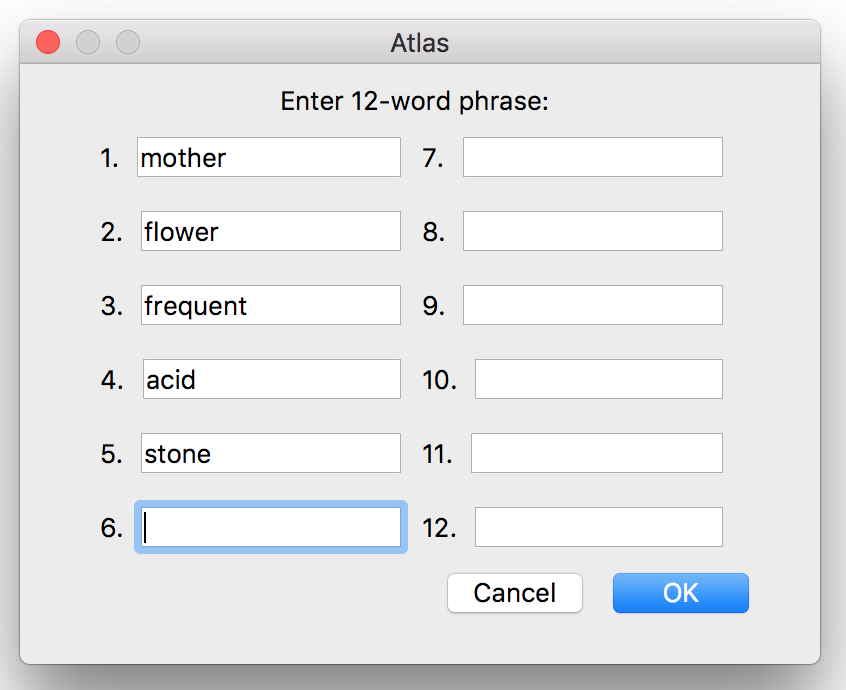
##### Wallet Creation

Upon starting the wallet, the user is prompted with a choice to either start a new wallet or restore an existing wallet. When a user starts a new wallet, a new seed is created that maintains the wallet. When a user chooses to restore a wallet, Atlas prompts the user for twelve words that comprises the mnemonic phrase. The user should keep these mnemonic words secret as they are the key to the wallet.



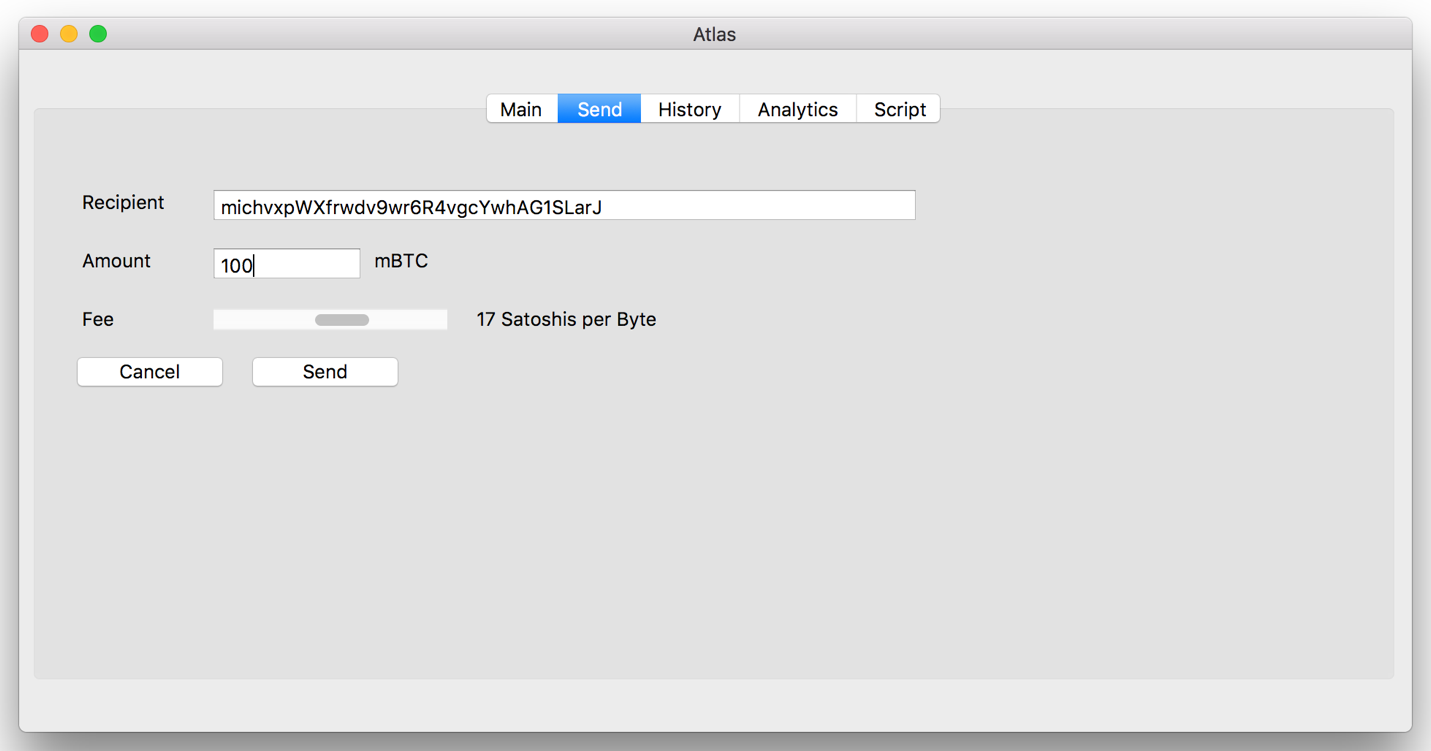
##### Mnemonic Phrase

The mnemonic phrase is a set of twelve words that create the wallet and make the wallet unique. This phrase should be kept secret for security. In its current implementation, the user cannot export their phrase. For development, a mnemonic phrase is included in the documentation in the Test Cases section. When starting the wallet, simply enter the 12 words into the boxes and it will restore an existing development wallet.



##### Sending Bitcoin

Sending a bitcoin transaction is very straightforward. A transaction in the Send tab allows the user to construct a basic transaction. A recipient represents another bitcoin payment address that the user will send funds to. The amount represents the number of bitcoins to send to the recipient. It is denominated in mBTC. The fees slider allows the user to change the amount of satoshis paid for the transaction fee. The user can have a minimum of zero satoshis per byte fees. It is important to note that Atlas sets a maximum fee which corresponds to the fee for the fastest transaction in the Analytics tab. This is implemented so that the user cannot overspend in fees.

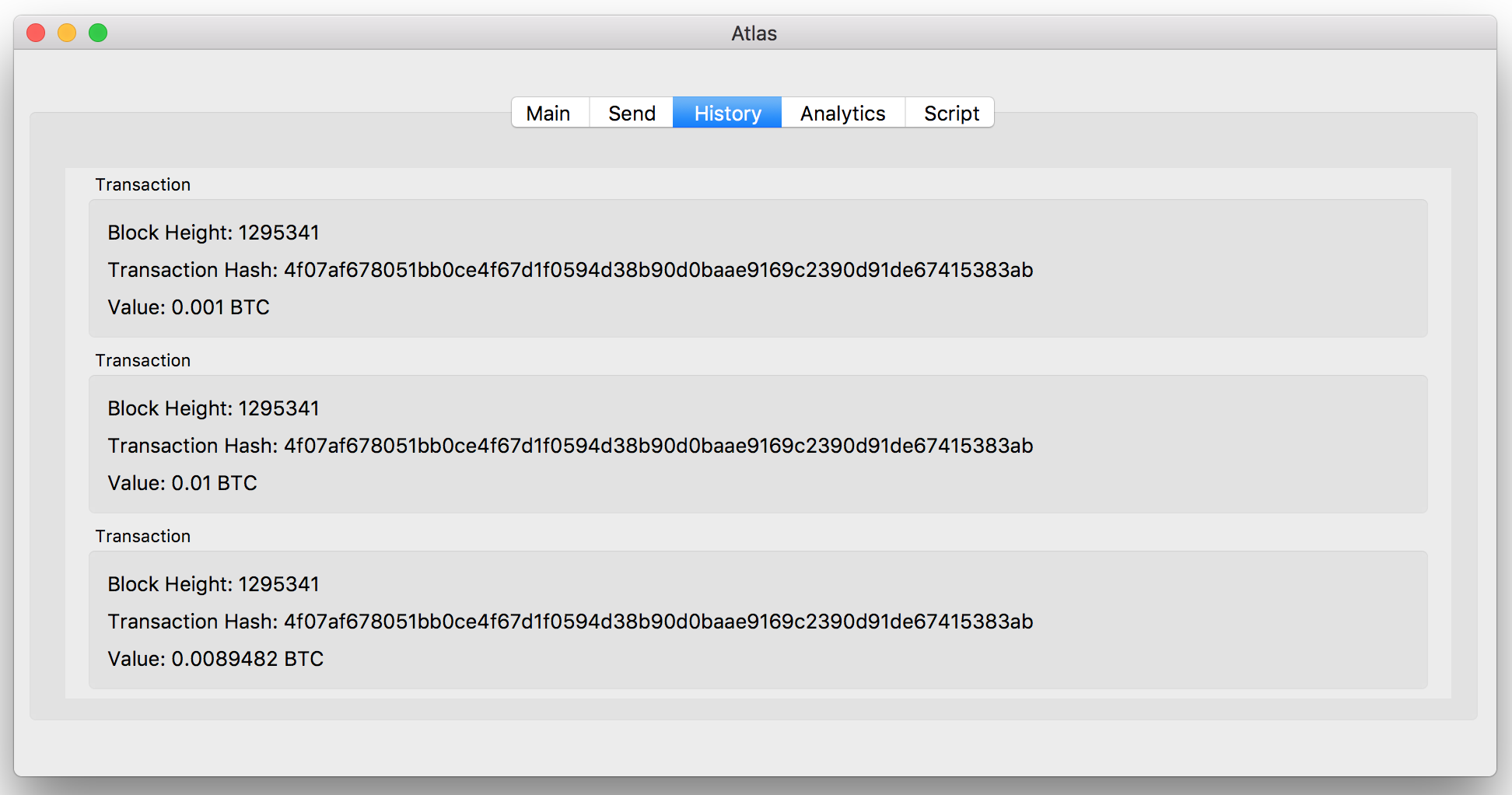


Below is a chart of bitcoin denominations for reference. It is from the Bitcoin wiki.



##### Transaction History

The History tab includes a list of transactions done by the addresses associated with the wallet. Each box in scroll area includes a transaction with a block height, transaction hash, and value. The block height represents where in the blockchain the transaction was confirmed, or mined. The transaction hash is a unique identifier for the transaction that the user can later reference. The value is the value of bitcoin transacted in that transaction.

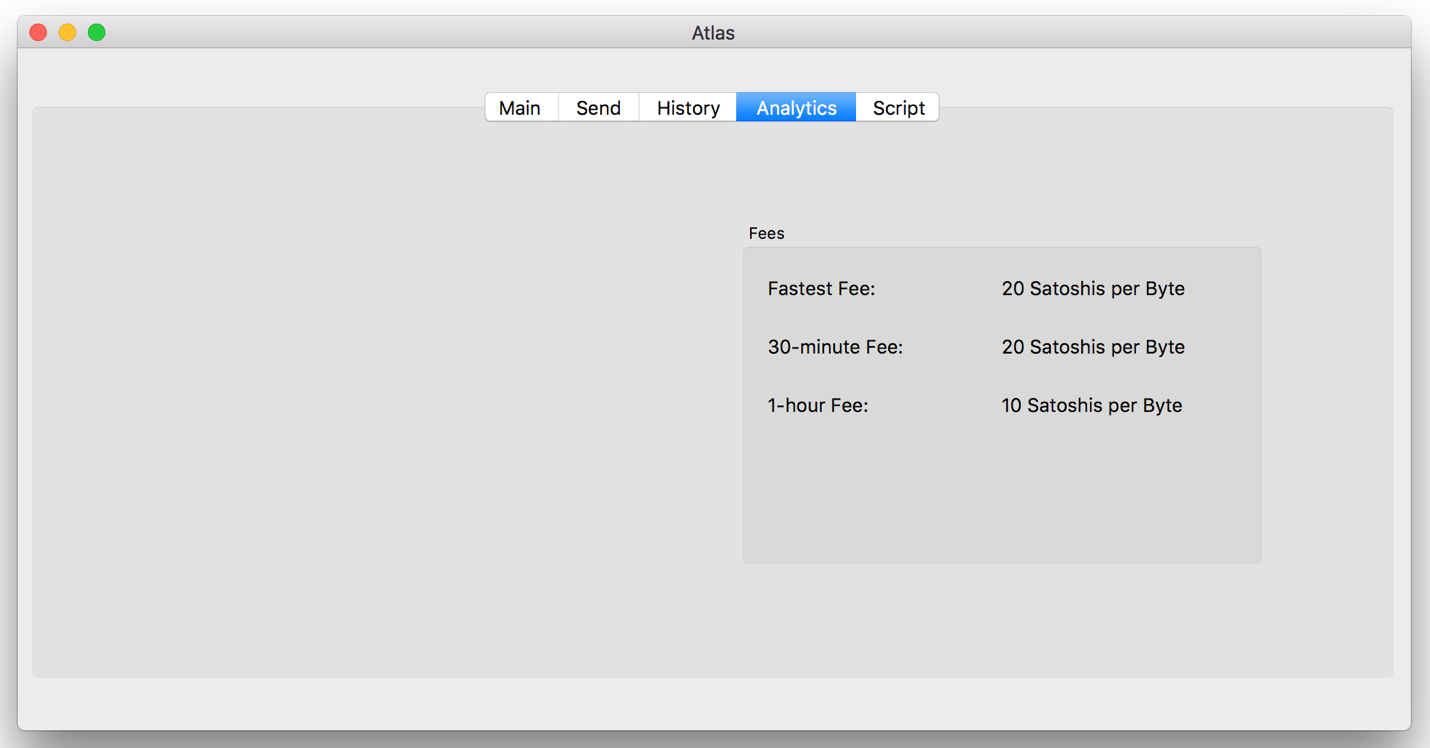


##### Understanding Network Fees

Transactions sent through the Bitcoin network may include a fee. The fee is optional and is set by the sender. A higher fee signals to the network that a specific transaction should be given priority for confirmation while a lower fee can lead to longer confirmation times. The fee market in Bitcoin is a free market set by supply and demand. As a result, the fee market changes over time and transactions can cost differently over the course of a day.

Atlas in its current implementation uses bitcoinfees.earn.com, a third-party API, to receive a suggested transaction fee. Using this API, Atlas is able to suggest to the user three fee costs located in the Analytics tab of Atlas. The user can choose to ignore these suggestions and selected a different fee.

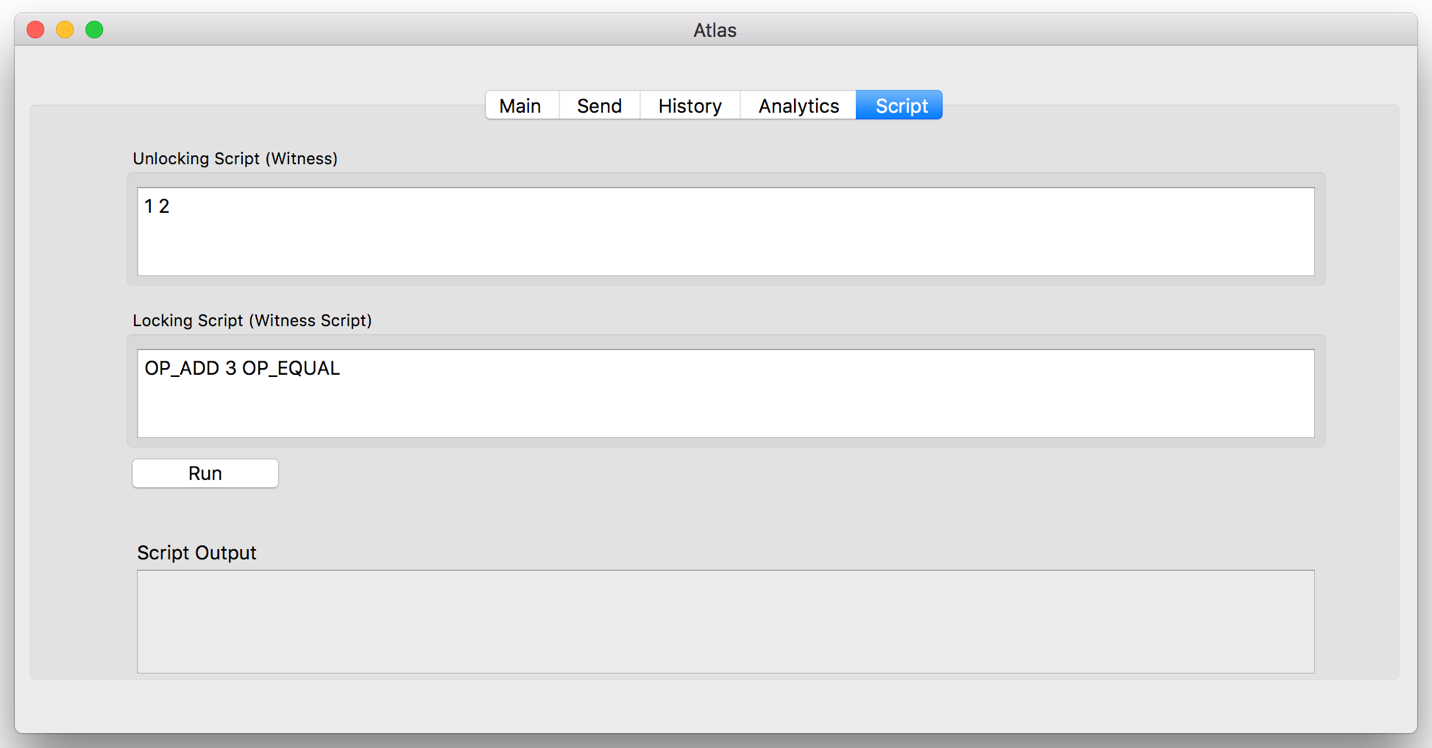
Fees are represented as Satoshi per Byte. This measurement informs the user that it costs n Satoshi per byte. If a transaction is 250 bytes in size and a suggested fee is 40 Satoshis per Byte, the fee will be 250\*40 Satoshis.



##### Using Bitcoin Script

Bitcoin protocol uses a stack-based programming language called Script that allows the user to create smart contracts and develop on the Bitcoin protocol. The Script tab in Atlas provides a basic interface for the user to learn and engage with Bitcoin Script by validating scripts. Witness and Witness Script serve as text edits where the user can write a basic script.

In this example, 1 and 2 are pushed onto the stack. Operation code OP\_ADD pops two items off the stack, adds them, and pushes the result onto the stack. Next, 3 is pushed onto the stack. Operation code OP\_EQUAL evaluates that the two top items are equal. Given that they are equal in this case, a True Boolean is pushed onto the stack and the stack executes successfully. With only a True Boolean left on the stack, the script is valid.



Below is a list of available operation codes that can be used in the Atlas script interpreter. The definitions for each operation code is from Antonopoulos’ *Mastering Bitcoin.*

|  |  |
| --- | --- |
| Available Operation Codes |  |
| OP\_DROP | Pop the top item in the stack. |
| OP\_DUP | Duplicate the top item in the stack. |
| OP\_DEPTH | Count the items on the stack and push the resulting count. |
| OP\_EQUAL | Push TRUE (1) if top two items are exactly equal, push FALSE (0) otherwise. |
| OP\_1ADD | Add 1 to the top item. |
| OP\_1SUB | Subtract 1 from the top item. |
| OP\_NEGATE | Flip the sign of the top item. |
| OP\_ABS | Change the sign of the top item to positive. |
| OP\_ADD | Pop top two items, add them and push result. |
| OP\_SUB | Pop top two items, subtract first from second, push result. |
| OP\_NUMEQUAL | Return TRUE if top two items are equal numbers. |
| OP\_NUMNOTEQUAL | Return TRUE if top two items are not equal numbers. |
| OP\_LESSTHAN | Return TRUE if second item is less than top item. |
| OP\_GREATERTHAN | Return TRUE if second item is greater than top item. |
| OP\_LESSTHANOREQUAL | Return TRUE if second item is less than or equal to top item. |
| OP\_GREATERTHANOREQUAL | Return TRUE if second item is greater than or equal to top item. |
| OP\_MIN | Return the smaller item of the two top items. |
| OP\_MAX | Return the larger of the two top items. |
| OP\_WITHIN | Return TRUE if the third item is between the second item (or equal) and first item. |
| OP\_RIPEMD160 | Return RIPEMD160 hash of top item. |
| OP\_SHA1 | Return SHA1 hash of top item. |
| OP\_SHA256 | Return SHA256 hash of top item. |
| OP\_HASH160 | Return RIPEMD160(SHA256(x)) hash of top item. |
| OP\_HASH256 | Return SHA256(SHA256(x)) hash of top item. |

The script output section of this tab serves as a console for the user where it is printed if a script is valid or not.

# Design

##### Philosophy

The wallet is designed in object-oriented principles. Objects are modularized and data is encapsulated in order to preserved object-oriented principles. The wallet is organized into basic wallet functionality, Bitcoin script manipulation, network functionality, utilities, and the front-end application.

##### Components

### List of high-level components that make the wallet fundamentals.

Base Wallet

|  |  |
| --- | --- |
| Components | Description |
| Mnemonic Code Words | Following a wallet standard, generated entropy will translate to 12 English words from a set. These words in addition to salt, will lead to a seed that creates a unique wallet. These 12 words could be written down and entered into the wallet to create this same unique wallet. |
| HD Wallet | A HD wallet, or deterministic wallet, is a wallet that creates a keychain based on a 512 bit seed. This is a standard in current Bitcoin wallets. |
| Bloom Filter | Bloom Filter is a standard privacy feature that allows the user to query for transactions without revealing to the network the specific transactions that he/she is asking for. |
| Peer Networking | Wallet connects to peers, does hand-shaking, and ask for transaction data. Most of the low-level work is handled by Libbitcoin library. |
| Payments | Allow user to send bitcoin and see the amount of bitcoin received. Allow user to generate new unique address for when receiving bitcoin |
| Graphical User Interface | Cross-platform Qt5 GUI. |

Analytics

|  |  |
| --- | --- |
| Components | Description |
| Fee Estimation | Query network to determine low, high, and median transaction fees. Provide recommendation to user for fee cost and when to send transaction. |
| Spend Analysis | Provide user with information on where bitcoin have been spent. |

Script

|  |  |
| --- | --- |
| Components | Description |
| Console | User can write their own bitcoin scripts and send them as transactions. Bitcoin Script language is stack-based language with limited OP codes. |
| Script debugger | Include debugger to help catch any errors in the user’s script. |

##### Classes

List of classes for backend and frontend of Atlas.

|  |  |
| --- | --- |
| Network Classes |  |
| Network | * Provides network functionality to the wallet including broadcasting transactions, reading data from Bitcoin blockchain, and accessing the bitcoinfees.earn.com API for transaction fee recommendations. * Utility functions for accessing fee recommendations. |
| Bloomfilter | * Privacy feature for querying network for inputs in a transaction including unspent transaction outputs (utxo). |

|  |  |
| --- | --- |
| Script |  |
| Script | * Provides functionality of Bitcoin script by simulating operations on a stack. * Includes a stack that serves as the execution stack for Bitcoin script. |
| Operation | * Includes operation codes and their functionality. Each operation code performs a function on the provided execution stack. * Includes utility cryptographic functions that can be used on the Bitcoin execution stack. |

|  |  |
| --- | --- |
| Utility |  |
| Valid\_Address | * Utility for validating if a string matches the consensus rules of the Bitcoin blockchain. |

|  |  |
| --- | --- |
| Wallet |  |
| Wallet | * Creates wallet seed. * Responsible for key management. * Responsible for address management. |
| utxo | * Manages the wallet’s record of unspent transaction outputs. |
| Transaction | * Provides utility functions for building a transaction. * Functionality to broadcast transaction when connected to the network. * Holds a history of previous transactions related to the wallet. |
| Error | * Handles exceptions and has an error stack. * Provides error log for debugging. |

|  |  |
| --- | --- |
| Qt |  |
| App | * User interface for the main wallet application. * User interface is divided into four tabs. |
| Restore\_wallet | * User interface for mnemonic prase input. * Validation of user input. |
| Start\_menu | * User interface for selecting to start a new wallet or restore an existing wallet. |

##### Data Structures

Important data structures that provides functionality for wallet fundamentals.

|  |  |
| --- | --- |
| Data Structures |  |
| feeEstimation | * Struct stores three fee recommendations to the user. * Contains satoshi value for fees that will result in transaction confirmation in fastest time, 30 minutes, or 60 minutes. |
| m\_ErrorMsgs | * Error stack that contains any errors that occurred within the wallet during runtime. |
| m\_tx | * Tuple data type representing any spent and confirmed transaction. * Contains satoshi value of transaction, transaction hash, and block height of the confirmed transaction. |
| m\_utxo | * Vector of tuples holding all unspent transaction outputs. These transactions outputs are spendable by the wallet. * Tuple contains satoshi value of transaction, transaction hash, and Bitcoin payment address. |

# Insights and Challenges

##### UTXO Management

The management of each transaction under the hood was more difficult than planned. The function of a wallet is to create an efficient manner for organizing transactions. Each payment address might have several different transactions associated with it. Each transaction will have varying values of bitcoins. The aim of any wallet is to provide a way to send bitcoin from any previous transaction. This becomes complicated fairly quickly as organizing transactions has a direct affect on the fees that the user pays. If an outgoing transaction relies on many input transactions, the fees paid by the user will be higher because the transaction size increases. Atlas uses a basic algorithm for sorting unspent transaction outputs on value. Going forward, a weighted approach will be needed so that value and number of inputs can be taken into account.

##### Transaction Building

Atlas currently only builds transactions that are pay-to-public-key-hash (P2PKH) which constitutes more than 80% of all transactions on the bitcoin network. The aim of Atlas was to experiment with building transactions that are more complex and less prevalent. There is still significant work to be done on transaction building as well as more efficient organization on how transactions are built.

# Conclusion

Writing this program was intensive ultimately I am still not satisfied with the product and will have to continue to update the project. There are several features that must be added in order for the wallet to become reliable and be able to use real Bitcoin funds. In this section, I will review the overall opportunities gained in designing the wallet and what I plan to add to it.

##### Opportunities

The opportunities in education gained by building a bitcoin wallet are very meaningful. The Bitcoin protocol has been in uninterrupted operation for nine years and does not show signs in losing relevancy. The venture capital money, developer interest, and philosophical intrigue into Bitcoin makes it worthwhile for at least some brief interest for any computer scientist.

The motivation in building Atlas were found in a desire to better understand the Bitcoin protocol. Most Bitcoin users interact with the protocol through the wallet abstraction layer, therefore understanding the mechanics of this software can allow a developer to significantly improve the way in which people interact with the protocol. Atlas was designed as an educational product that I hope to soon develop into an industry level product. The exciting challenges in learning how transactions and protocol work has also invited new ideas to explore going forward.

On a technical level, designing Atlas has made me more comfortable in designing large programs as well as have a better understanding on how to design object-oriented code. Atlas is not a perfect example of object-oriented principles, but I now know the underlying issues in order to challenge the code base and make it near-perfect.

The C++ language for this project for its object-oriented design and flexibility of memory management. The language is cross-platform which allows for flexibility on the devices that can run it. The Libbitcoin bitcoin development library was very resourceful in designing the wallet because it was able to abstract several low-level cryptography and functionality. Boost library was also resourceful in using property trees data structure. The C++ is time-tested, has significant developer resources, and is overall a very flexible language that gives the programmer a large amount of control over detail.

Using the Libbitcoin library API was very challenging initially due to the scarce resources available for its latest version. However, the documentation is growing and being updated more consistently since starting Atlas.

Bitcoin’s ecosystem is nascent, but there are several industry standards that were used in the design of Atlas. In particular, BIP 39 for mnemonic phrase and BIP 32 for the hierarchical design of the wallet’s keychain.

Designing Atlas required a large learning curve about Bitcoin and a basic understanding of its cryptography. Andreas Antonopoulos’ *Mastering Bitcoin* was incredibly resourceful that provided a strong technical foundation in Bitcoin’s protocol.

The graphical user interface of Atlas was written in Qt5 which is a cross-platform library for cross-platform applications. Qt5 has both extensive documentation and very reliable codebase for a free product. As part of using the library, it was agreed that Atlas will remain open-source.

##### Next Steps

There are several next steps to Atlas in order to improve the functionality and reliability of the application. In particular, BIP 21 will be implemented in order to allow QR codes to be presented. In addition, better fee recommendations and analytics will need to be added in order for the user to have a better understanding of their funds. In the initial proposal of the wallet, providing analytics was a major pillar of the application. Unfortunately, Atlas only provides fee estimation through a 3rd party API. Going forward, it must not rely on the API and instead use dynamic fee estimation. In addition to fee estimation, an analytics dashboard will be implemented that shows where and how bitcoin are spent.

There is still significant work to be done for the user’s exposure to the Bitcoin script language. In its current form, the user can only test and debug a Bitcoin script. Going forward, the user should be provided a way to submit transactions with their written script. In addition, after conversations with helpful developers as well as reading supplementary material, there are new ways in developing a more user-friendly approach to script construction that I will have to implement.

# Bibliography

The following resources were very helpful in building Atlas.

Antonopoulos, Andreas M. *Mastering Bitcoin: Programming the Open Blockchain.* 2nd ed., O'Reilly Media, 2017.

Jaramillo, Aaron. “The Libbitcoin Tutorials.” *The Web Log of Aaron Jaramillo*, aaronjaramillo.org/category/libbitcoindocs.

“Libbitcoin Wiki.” *GitHub, Github*, github.com/libbitcoin/libbitcoin/wiki.

“Libbitcoin Documentation.” *Overview - Libbitcoin 1 Documentation*, libbitcoin.dyne.org/doc/overview.html.

# Source Code

~~~~~~~~~~~~~~~~~~~~~Source code for file network.cpp~~~~~~~~~~~~~~~~~~~~~

#include "../wallet/stdafx.h"

Network::Network()

{

m\_client = NULL;

m\_fees = new feeEstimation;

}

Network::~Network()

{

delete m\_client;

delete m\_fees;

}

/\*\*

\* @brief Returns obelisk client pointer that allows rpc calls to be done.

\*

\* @return bc::client::obelisk\_client&

\*/

bc::client::obelisk\_client& Network::connect()

{

// Testnet connection details.

bc::client::connection\_type connection = {};

connection.retries = 3;

connection.timeout\_seconds = 8;

connection.server = bc::config::endpoint("tcp://testnet3.libbitcoin.net:19091");

//TODO Timeouts?

//List of servers: https://github.com/libbitcoin/libbitcoin-server/wiki/Community-Servers

// Initialize obelisk.

m\_client = new bc::client::obelisk\_client(connection);

// Check if connection is working.

if(m\_client->connect(connection))

{

std::cout << "Connected to Libbitcoin.net" << std::endl;

return \*m\_client;

}

else

{

Error::RecordError(std::string("Error connecting to bitcoin network."));

// should probably return something else.

return \*m\_client;

}

};

bool Network::disconnect()

{

std::cout << "Disconnected from Libbitcoin.net" << std::endl;

delete m\_client;

m\_client = NULL;

};

// from stackoverflow

std::size\_t callback(const char\* in, std::size\_t size, std::size\_t num, std::string\* out)

{

const std::size\_t totalBytes(size \* num);

out->append(in, totalBytes);

return totalBytes;

};

void Network::refreshFeeRecommendations()

{

// Instantiate curl objects.

CURL \*curl;

CURLcode res;

std::string buffer;

// Init curl.

curl\_global\_init(CURL\_GLOBAL\_DEFAULT);

curl = curl\_easy\_init();

if(curl)

{

// Request fee recommendations from bitcoinfees.earn.com (trusted recommendation).

// TODO: internal free recommendation tool.

curl\_easy\_setopt(curl, CURLOPT\_URL, "https://bitcoinfees.earn.com/api/v1/fees/recommended");

// Timeout after 10 seconds.

curl\_easy\_setopt(curl, CURLOPT\_TIMEOUT, 10);

std::unique\_ptr<std::string> httpData(new std::string());

// Response.

curl\_easy\_setopt(curl, CURLOPT\_WRITEFUNCTION, callback);

curl\_easy\_setopt(curl, CURLOPT\_WRITEDATA, httpData.get());

res = curl\_easy\_perform(curl);

// Error checking.

if(res != CURLE\_OK)

{

fprintf(stderr, "curl\_easy\_perform() failed: %s\n", curl\_easy\_strerror(res));

}

else

{

Json::Value jsonData;

Json::Reader jsonReader;

// Parse JSON.

if (jsonReader.parse(\*httpData, jsonData))

{

std::cout << "Successfully parsed JSON data" << std::endl;

std::cout << "\nJSON data received:" << std::endl;

std::cout << jsonData.toStyledString() << std::endl;

// Load fees into feeEstimation struct.

m\_fees -> fastestFee = jsonData["fastestFee"].asUInt64();

m\_fees -> halfHourFee = jsonData["halfHourFee"].asUInt64();

m\_fees -> hourFee = jsonData["hourFee"].asUInt64();

std::cout<<m\_fees<<std::endl;

};

};

// Clean-up.

curl\_easy\_cleanup(curl);

};

// Clean-up

curl\_global\_cleanup();

};

~~~~~~~~~~~~~~~~~~~~~Source code for file bloomfilter.cpp~~~~~~~~~~~~~~~~~~~~~

#include "stdafx.h"

#include "BloomFilter.h"

#include "Error.h"

/\*\*/

/\*

BloomFilter::bloomFilterHash()

NAME

BloomFilter::bloomFilterHash()

SYNOPSIS

void BloomFilter::bloomFilterHash()

DESCRIPTION

Creates hash for bloom filter.

RETURNS

Returns murmur3 hash.

AUTHOR

Philip Glazman

DATE

1/11/2018

\*/

/\*\*/

void BloomFilter::bloomFilterHash()

{

//nHashNum \* 0xFBA4C795 + nTweak

int murmurSeed = 0xFBA4C795 + m\_nTweak;

//TODO import murmur3 from hash.h (Bitcoin/Bitcoin)

}

~~~~~~~~~~~~~~~~~~~~~Source code for file ui\_restore\_wallet.h~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'restore\_wallet.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.10.1

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef UI\_RESTORE\_WALLET\_H

#define UI\_RESTORE\_WALLET\_H

#include <QtCore/QVariant>

#include <QtWidgets/QAction>

#include <QtWidgets/QApplication>

#include <QtWidgets/QButtonGroup>

#include <QtWidgets/QDialog>

#include <QtWidgets/QDialogButtonBox>

#include <QtWidgets/QHBoxLayout>

#include <QtWidgets/QHeaderView>

#include <QtWidgets/QLabel>

#include <QtWidgets/QLineEdit>

#include <QtWidgets/QVBoxLayout>

#include <QtWidgets/QWidget>

QT\_BEGIN\_NAMESPACE

class Ui\_restore\_wallet

{

public:

QDialogButtonBox \*buttonBox;

QLabel \*label\_13;

QWidget \*widget;

QHBoxLayout \*horizontalLayout\_13;

QVBoxLayout \*verticalLayout;

QHBoxLayout \*horizontalLayout;

QLabel \*label;

QLineEdit \*word\_1;

QHBoxLayout \*horizontalLayout\_2;

QLabel \*label\_2;

QLineEdit \*word\_2;

QHBoxLayout \*horizontalLayout\_3;

QLabel \*label\_3;

QLineEdit \*word\_3;

QHBoxLayout \*horizontalLayout\_4;

QLabel \*label\_4;

QLineEdit \*word\_4;

QHBoxLayout \*horizontalLayout\_5;

QLabel \*label\_5;

QLineEdit \*word\_5;

QHBoxLayout \*horizontalLayout\_6;

QLabel \*label\_6;

QLineEdit \*word\_6;

QVBoxLayout \*verticalLayout\_2;

QHBoxLayout \*horizontalLayout\_7;

QLabel \*label\_7;

QLineEdit \*word\_7;

QHBoxLayout \*horizontalLayout\_8;

QLabel \*label\_8;

QLineEdit \*word\_8;

QHBoxLayout \*horizontalLayout\_9;

QLabel \*label\_9;

QLineEdit \*word\_9;

QHBoxLayout \*horizontalLayout\_10;

QLabel \*label\_10;

QLineEdit \*word\_10;

QHBoxLayout \*horizontalLayout\_11;

QLabel \*label\_11;

QLineEdit \*word\_11;

QHBoxLayout \*horizontalLayout\_12;

QLabel \*label\_12;

QLineEdit \*word\_12;

void setupUi(QDialog \*restore\_wallet)

{

if (restore\_wallet->objectName().isEmpty())

restore\_wallet->setObjectName(QStringLiteral("restore\_wallet"));

restore\_wallet->resize(400, 300);

buttonBox = new QDialogButtonBox(restore\_wallet);

buttonBox->setObjectName(QStringLiteral("buttonBox"));

buttonBox->setGeometry(QRect(30, 250, 341, 32));

buttonBox->setOrientation(Qt::Horizontal);

buttonBox->setStandardButtons(QDialogButtonBox::Cancel|QDialogButtonBox::Ok);

buttonBox->setCenterButtons(false);

label\_13 = new QLabel(restore\_wallet);

label\_13->setObjectName(QStringLiteral("label\_13"));

label\_13->setGeometry(QRect(130, 10, 141, 16));

widget = new QWidget(restore\_wallet);

widget->setObjectName(QStringLiteral("widget"));

widget->setGeometry(QRect(40, 30, 312, 218));

widget->setAutoFillBackground(false);

widget->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_13 = new QHBoxLayout(widget);

horizontalLayout\_13->setObjectName(QStringLiteral("horizontalLayout\_13"));

horizontalLayout\_13->setContentsMargins(0, 0, 0, 0);

verticalLayout = new QVBoxLayout();

verticalLayout->setObjectName(QStringLiteral("verticalLayout"));

horizontalLayout = new QHBoxLayout();

horizontalLayout->setObjectName(QStringLiteral("horizontalLayout"));

label = new QLabel(widget);

label->setObjectName(QStringLiteral("label"));

label->setAutoFillBackground(false);

label->setInputMethodHints(Qt::ImhNone);

horizontalLayout->addWidget(label);

word\_1 = new QLineEdit(widget);

word\_1->setObjectName(QStringLiteral("word\_1"));

word\_1->setAutoFillBackground(false);

word\_1->setInputMethodHints(Qt::ImhNone);

word\_1->setMaxLength(50);

word\_1->setClearButtonEnabled(false);

horizontalLayout->addWidget(word\_1);

verticalLayout->addLayout(horizontalLayout);

horizontalLayout\_2 = new QHBoxLayout();

horizontalLayout\_2->setObjectName(QStringLiteral("horizontalLayout\_2"));

label\_2 = new QLabel(widget);

label\_2->setObjectName(QStringLiteral("label\_2"));

label\_2->setAutoFillBackground(false);

label\_2->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_2->addWidget(label\_2);

word\_2 = new QLineEdit(widget);

word\_2->setObjectName(QStringLiteral("word\_2"));

word\_2->setAutoFillBackground(false);

word\_2->setInputMethodHints(Qt::ImhNone);

word\_2->setMaxLength(50);

word\_2->setClearButtonEnabled(false);

horizontalLayout\_2->addWidget(word\_2);

verticalLayout->addLayout(horizontalLayout\_2);

horizontalLayout\_3 = new QHBoxLayout();

horizontalLayout\_3->setObjectName(QStringLiteral("horizontalLayout\_3"));

label\_3 = new QLabel(widget);

label\_3->setObjectName(QStringLiteral("label\_3"));

label\_3->setAutoFillBackground(false);

label\_3->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_3->addWidget(label\_3);

word\_3 = new QLineEdit(widget);

word\_3->setObjectName(QStringLiteral("word\_3"));

word\_3->setAutoFillBackground(false);

word\_3->setInputMethodHints(Qt::ImhNone);

word\_3->setMaxLength(50);

word\_3->setClearButtonEnabled(false);

horizontalLayout\_3->addWidget(word\_3);

verticalLayout->addLayout(horizontalLayout\_3);

horizontalLayout\_4 = new QHBoxLayout();

horizontalLayout\_4->setObjectName(QStringLiteral("horizontalLayout\_4"));

label\_4 = new QLabel(widget);

label\_4->setObjectName(QStringLiteral("label\_4"));

label\_4->setAutoFillBackground(false);

label\_4->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_4->addWidget(label\_4);

word\_4 = new QLineEdit(widget);

word\_4->setObjectName(QStringLiteral("word\_4"));

word\_4->setAutoFillBackground(false);

word\_4->setInputMethodHints(Qt::ImhNone);

word\_4->setMaxLength(50);

word\_4->setClearButtonEnabled(false);

horizontalLayout\_4->addWidget(word\_4);

verticalLayout->addLayout(horizontalLayout\_4);

horizontalLayout\_5 = new QHBoxLayout();

horizontalLayout\_5->setObjectName(QStringLiteral("horizontalLayout\_5"));

label\_5 = new QLabel(widget);

label\_5->setObjectName(QStringLiteral("label\_5"));

label\_5->setAutoFillBackground(false);

label\_5->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_5->addWidget(label\_5);

word\_5 = new QLineEdit(widget);

word\_5->setObjectName(QStringLiteral("word\_5"));

word\_5->setAutoFillBackground(false);

word\_5->setInputMethodHints(Qt::ImhNone);

word\_5->setMaxLength(50);

word\_5->setClearButtonEnabled(false);

horizontalLayout\_5->addWidget(word\_5);

verticalLayout->addLayout(horizontalLayout\_5);

horizontalLayout\_6 = new QHBoxLayout();

horizontalLayout\_6->setObjectName(QStringLiteral("horizontalLayout\_6"));

label\_6 = new QLabel(widget);

label\_6->setObjectName(QStringLiteral("label\_6"));

label\_6->setAutoFillBackground(false);

label\_6->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_6->addWidget(label\_6);

word\_6 = new QLineEdit(widget);

word\_6->setObjectName(QStringLiteral("word\_6"));

word\_6->setAutoFillBackground(false);

word\_6->setInputMethodHints(Qt::ImhNone);

word\_6->setMaxLength(50);

word\_6->setClearButtonEnabled(false);

horizontalLayout\_6->addWidget(word\_6);

verticalLayout->addLayout(horizontalLayout\_6);

horizontalLayout\_13->addLayout(verticalLayout);

verticalLayout\_2 = new QVBoxLayout();

verticalLayout\_2->setObjectName(QStringLiteral("verticalLayout\_2"));

horizontalLayout\_7 = new QHBoxLayout();

horizontalLayout\_7->setObjectName(QStringLiteral("horizontalLayout\_7"));

label\_7 = new QLabel(widget);

label\_7->setObjectName(QStringLiteral("label\_7"));

label\_7->setAutoFillBackground(false);

label\_7->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_7->addWidget(label\_7);

word\_7 = new QLineEdit(widget);

word\_7->setObjectName(QStringLiteral("word\_7"));

word\_7->setAutoFillBackground(false);

word\_7->setInputMethodHints(Qt::ImhNone);

word\_7->setMaxLength(50);

word\_7->setClearButtonEnabled(false);

horizontalLayout\_7->addWidget(word\_7);

verticalLayout\_2->addLayout(horizontalLayout\_7);

horizontalLayout\_8 = new QHBoxLayout();

horizontalLayout\_8->setObjectName(QStringLiteral("horizontalLayout\_8"));

label\_8 = new QLabel(widget);

label\_8->setObjectName(QStringLiteral("label\_8"));

label\_8->setAutoFillBackground(false);

label\_8->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_8->addWidget(label\_8);

word\_8 = new QLineEdit(widget);

word\_8->setObjectName(QStringLiteral("word\_8"));

word\_8->setAutoFillBackground(false);

word\_8->setInputMethodHints(Qt::ImhNone);

word\_8->setMaxLength(50);

word\_8->setClearButtonEnabled(false);

horizontalLayout\_8->addWidget(word\_8);

verticalLayout\_2->addLayout(horizontalLayout\_8);

horizontalLayout\_9 = new QHBoxLayout();

horizontalLayout\_9->setObjectName(QStringLiteral("horizontalLayout\_9"));

label\_9 = new QLabel(widget);

label\_9->setObjectName(QStringLiteral("label\_9"));

label\_9->setAutoFillBackground(false);

label\_9->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_9->addWidget(label\_9);

word\_9 = new QLineEdit(widget);

word\_9->setObjectName(QStringLiteral("word\_9"));

word\_9->setAutoFillBackground(false);

word\_9->setInputMethodHints(Qt::ImhNone);

word\_9->setMaxLength(50);

word\_9->setClearButtonEnabled(false);

horizontalLayout\_9->addWidget(word\_9);

verticalLayout\_2->addLayout(horizontalLayout\_9);

horizontalLayout\_10 = new QHBoxLayout();

horizontalLayout\_10->setObjectName(QStringLiteral("horizontalLayout\_10"));

label\_10 = new QLabel(widget);

label\_10->setObjectName(QStringLiteral("label\_10"));

label\_10->setAutoFillBackground(false);

label\_10->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_10->addWidget(label\_10);

word\_10 = new QLineEdit(widget);

word\_10->setObjectName(QStringLiteral("word\_10"));

word\_10->setAutoFillBackground(false);

word\_10->setInputMethodHints(Qt::ImhNone);

word\_10->setMaxLength(50);

word\_10->setClearButtonEnabled(false);

horizontalLayout\_10->addWidget(word\_10);

verticalLayout\_2->addLayout(horizontalLayout\_10);

horizontalLayout\_11 = new QHBoxLayout();

horizontalLayout\_11->setObjectName(QStringLiteral("horizontalLayout\_11"));

label\_11 = new QLabel(widget);

label\_11->setObjectName(QStringLiteral("label\_11"));

label\_11->setAutoFillBackground(false);

label\_11->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_11->addWidget(label\_11);

word\_11 = new QLineEdit(widget);

word\_11->setObjectName(QStringLiteral("word\_11"));

word\_11->setAutoFillBackground(false);

word\_11->setInputMethodHints(Qt::ImhNone);

word\_11->setMaxLength(50);

word\_11->setClearButtonEnabled(false);

horizontalLayout\_11->addWidget(word\_11);

verticalLayout\_2->addLayout(horizontalLayout\_11);

horizontalLayout\_12 = new QHBoxLayout();

horizontalLayout\_12->setObjectName(QStringLiteral("horizontalLayout\_12"));

label\_12 = new QLabel(widget);

label\_12->setObjectName(QStringLiteral("label\_12"));

label\_12->setAutoFillBackground(false);

label\_12->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_12->addWidget(label\_12);

word\_12 = new QLineEdit(widget);

word\_12->setObjectName(QStringLiteral("word\_12"));

word\_12->setAutoFillBackground(false);

word\_12->setInputMethodHints(Qt::ImhNone);

word\_12->setMaxLength(50);

word\_12->setClearButtonEnabled(false);

horizontalLayout\_12->addWidget(word\_12);

verticalLayout\_2->addLayout(horizontalLayout\_12);

horizontalLayout\_13->addLayout(verticalLayout\_2);

retranslateUi(restore\_wallet);

QObject::connect(buttonBox, SIGNAL(accepted()), restore\_wallet, SLOT(accept()));

QObject::connect(buttonBox, SIGNAL(rejected()), restore\_wallet, SLOT(reject()));

QMetaObject::connectSlotsByName(restore\_wallet);

} // setupUi

void retranslateUi(QDialog \*restore\_wallet)

{

restore\_wallet->setWindowTitle(QApplication::translate("restore\_wallet", "Dialog", nullptr));

label\_13->setText(QApplication::translate("restore\_wallet", "Enter 12-word phrase:", nullptr));

label->setText(QApplication::translate("restore\_wallet", "1.", nullptr));

word\_1->setPlaceholderText(QString());

label\_2->setText(QApplication::translate("restore\_wallet", "2.", nullptr));

word\_2->setPlaceholderText(QString());

label\_3->setText(QApplication::translate("restore\_wallet", "3.", nullptr));

word\_3->setPlaceholderText(QString());

label\_4->setText(QApplication::translate("restore\_wallet", "4.", nullptr));

word\_4->setPlaceholderText(QString());

label\_5->setText(QApplication::translate("restore\_wallet", "5.", nullptr));

word\_5->setPlaceholderText(QString());

label\_6->setText(QApplication::translate("restore\_wallet", "6.", nullptr));

word\_6->setPlaceholderText(QString());

label\_7->setText(QApplication::translate("restore\_wallet", "7.", nullptr));

word\_7->setPlaceholderText(QString());

label\_8->setText(QApplication::translate("restore\_wallet", "8.", nullptr));

word\_8->setPlaceholderText(QString());

label\_9->setText(QApplication::translate("restore\_wallet", "9.", nullptr));

word\_9->setPlaceholderText(QString());

label\_10->setText(QApplication::translate("restore\_wallet", "10.", nullptr));

word\_10->setPlaceholderText(QString());

label\_11->setText(QApplication::translate("restore\_wallet", "11.", nullptr));

word\_11->setPlaceholderText(QString());

label\_12->setText(QApplication::translate("restore\_wallet", "12.", nullptr));

word\_12->setPlaceholderText(QString());

} // retranslateUi

};

namespace Ui {

class restore\_wallet: public Ui\_restore\_wallet {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // UI\_RESTORE\_WALLET\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file ui\_start\_wallet.h~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'start\_wallet.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.10.1

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef UI\_START\_WALLET\_H

#define UI\_START\_WALLET\_H

#include <QtCore/QVariant>

#include <QtWidgets/QAction>

#include <QtWidgets/QApplication>

#include <QtWidgets/QButtonGroup>

#include <QtWidgets/QHeaderView>

#include <QtWidgets/QPushButton>

#include <QtWidgets/QVBoxLayout>

#include <QtWidgets/QWidget>

QT\_BEGIN\_NAMESPACE

class Ui\_start\_wallet

{

public:

QWidget \*layoutWidget;

QVBoxLayout \*verticalLayout;

QPushButton \*pushButton;

QPushButton \*restore\_existing\_wallet;

void setupUi(QWidget \*start\_wallet)

{

if (start\_wallet->objectName().isEmpty())

start\_wallet->setObjectName(QStringLiteral("start\_wallet"));

start\_wallet->resize(400, 300);

layoutWidget = new QWidget(start\_wallet);

layoutWidget->setObjectName(QStringLiteral("layoutWidget"));

layoutWidget->setGeometry(QRect(80, 80, 291, 111));

verticalLayout = new QVBoxLayout(layoutWidget);

verticalLayout->setObjectName(QStringLiteral("verticalLayout"));

verticalLayout->setContentsMargins(0, 0, 0, 0);

pushButton = new QPushButton(layoutWidget);

pushButton->setObjectName(QStringLiteral("pushButton"));

verticalLayout->addWidget(pushButton);

restore\_existing\_wallet = new QPushButton(layoutWidget);

restore\_existing\_wallet->setObjectName(QStringLiteral("restore\_existing\_wallet"));

verticalLayout->addWidget(restore\_existing\_wallet);

retranslateUi(start\_wallet);

QMetaObject::connectSlotsByName(start\_wallet);

} // setupUi

void retranslateUi(QWidget \*start\_wallet)

{

start\_wallet->setWindowTitle(QApplication::translate("start\_wallet", "Form", nullptr));

pushButton->setText(QApplication::translate("start\_wallet", "Create New Wallet", nullptr));

restore\_existing\_wallet->setText(QApplication::translate("start\_wallet", "Restore Existing Wallet", nullptr));

} // retranslateUi

};

namespace Ui {

class start\_wallet: public Ui\_start\_wallet {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // UI\_START\_WALLET\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file moc\_app.cpp~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Meta object code from reading C++ file 'app.h'

\*\*

\*\* Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)

\*\*

\*\* WARNING! All changes made in this file will be lost!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "../Atlas/app.h"

#include <QtCore/qbytearray.h>

#include <QtCore/qmetatype.h>

#if !defined(Q\_MOC\_OUTPUT\_REVISION)

#error "The header file 'app.h' doesn't include <QObject>."

#elif Q\_MOC\_OUTPUT\_REVISION != 67

#error "This file was generated using the moc from 5.10.1. It"

#error "cannot be used with the include files from this version of Qt."

#error "(The moc has changed too much.)"

#endif

QT\_BEGIN\_MOC\_NAMESPACE

QT\_WARNING\_PUSH

QT\_WARNING\_DISABLE\_DEPRECATED

struct qt\_meta\_stringdata\_app\_t {

QByteArrayData data[3];

char stringdata0[40];

};

#define QT\_MOC\_LITERAL(idx, ofs, len) \

Q\_STATIC\_BYTE\_ARRAY\_DATA\_HEADER\_INITIALIZER\_WITH\_OFFSET(len, \

qptrdiff(offsetof(qt\_meta\_stringdata\_app\_t, stringdata0) + ofs \

- idx \* sizeof(QByteArrayData)) \

)

static const qt\_meta\_stringdata\_app\_t qt\_meta\_stringdata\_app = {

{

QT\_MOC\_LITERAL(0, 0, 3), // "app"

QT\_MOC\_LITERAL(1, 4, 34), // "on\_restore\_existing\_wallet\_cl..."

QT\_MOC\_LITERAL(2, 39, 0) // ""

},

"app\0on\_restore\_existing\_wallet\_clicked\0"

""

};

#undef QT\_MOC\_LITERAL

static const uint qt\_meta\_data\_app[] = {

// content:

7, // revision

0, // classname

0, 0, // classinfo

1, 14, // methods

0, 0, // properties

0, 0, // enums/sets

0, 0, // constructors

0, // flags

0, // signalCount

// slots: name, argc, parameters, tag, flags

1, 0, 19, 2, 0x08 /\* Private \*/,

// slots: parameters

QMetaType::Void,

0 // eod

};

void app::qt\_static\_metacall(QObject \*\_o, QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

if (\_c == QMetaObject::InvokeMetaMethod) {

app \*\_t = static\_cast<app \*>(\_o);

Q\_UNUSED(\_t)

switch (\_id) {

case 0: \_t->on\_restore\_existing\_wallet\_clicked(); break;

default: ;

}

}

Q\_UNUSED(\_a);

}

QT\_INIT\_METAOBJECT const QMetaObject app::staticMetaObject = {

{ &QMainWindow::staticMetaObject, qt\_meta\_stringdata\_app.data,

qt\_meta\_data\_app, qt\_static\_metacall, nullptr, nullptr}

};

const QMetaObject \*app::metaObject() const

{

return QObject::d\_ptr->metaObject ? QObject::d\_ptr->dynamicMetaObject() : &staticMetaObject;

}

void \*app::qt\_metacast(const char \*\_clname)

{

if (!\_clname) return nullptr;

if (!strcmp(\_clname, qt\_meta\_stringdata\_app.stringdata0))

return static\_cast<void\*>(this);

return QMainWindow::qt\_metacast(\_clname);

}

int app::qt\_metacall(QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

\_id = QMainWindow::qt\_metacall(\_c, \_id, \_a);

if (\_id < 0)

return \_id;

if (\_c == QMetaObject::InvokeMetaMethod) {

if (\_id < 1)

qt\_static\_metacall(this, \_c, \_id, \_a);

\_id -= 1;

} else if (\_c == QMetaObject::RegisterMethodArgumentMetaType) {

if (\_id < 1)

\*reinterpret\_cast<int\*>(\_a[0]) = -1;

\_id -= 1;

}

return \_id;

}

QT\_WARNING\_POP

QT\_END\_MOC\_NAMESPACE

~~~~~~~~~~~~~~~~~~~~~Source code for file ui\_start\_menu.h~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'start\_menu.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.10.1

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef UI\_START\_MENU\_H

#define UI\_START\_MENU\_H

#include <QtCore/QVariant>

#include <QtWidgets/QAction>

#include <QtWidgets/QApplication>

#include <QtWidgets/QButtonGroup>

#include <QtWidgets/QDialog>

#include <QtWidgets/QHeaderView>

#include <QtWidgets/QPushButton>

#include <QtWidgets/QVBoxLayout>

#include <QtWidgets/QWidget>

QT\_BEGIN\_NAMESPACE

class Ui\_start\_menu

{

public:

QWidget \*layoutWidget;

QVBoxLayout \*verticalLayout;

QPushButton \*create\_new\_wallet;

QPushButton \*restore\_existing\_wallet;

void setupUi(QDialog \*start\_menu)

{

if (start\_menu->objectName().isEmpty())

start\_menu->setObjectName(QStringLiteral("start\_menu"));

start\_menu->resize(400, 300);

layoutWidget = new QWidget(start\_menu);

layoutWidget->setObjectName(QStringLiteral("layoutWidget"));

layoutWidget->setGeometry(QRect(40, 80, 291, 111));

verticalLayout = new QVBoxLayout(layoutWidget);

verticalLayout->setObjectName(QStringLiteral("verticalLayout"));

verticalLayout->setContentsMargins(0, 0, 0, 0);

create\_new\_wallet = new QPushButton(layoutWidget);

create\_new\_wallet->setObjectName(QStringLiteral("create\_new\_wallet"));

verticalLayout->addWidget(create\_new\_wallet);

restore\_existing\_wallet = new QPushButton(layoutWidget);

restore\_existing\_wallet->setObjectName(QStringLiteral("restore\_existing\_wallet"));

verticalLayout->addWidget(restore\_existing\_wallet);

retranslateUi(start\_menu);

QMetaObject::connectSlotsByName(start\_menu);

} // setupUi

void retranslateUi(QDialog \*start\_menu)

{

start\_menu->setWindowTitle(QApplication::translate("start\_menu", "Dialog", nullptr));

create\_new\_wallet->setText(QApplication::translate("start\_menu", "Create New Wallet", nullptr));

restore\_existing\_wallet->setText(QApplication::translate("start\_menu", "Restore Existing Wallet", nullptr));

} // retranslateUi

};

namespace Ui {

class start\_menu: public Ui\_start\_menu {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // UI\_START\_MENU\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file moc\_start\_menu.cpp~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Meta object code from reading C++ file 'start\_menu.h'

\*\*

\*\* Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)

\*\*

\*\* WARNING! All changes made in this file will be lost!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "../Atlas/start\_menu.h"

#include <QtCore/qbytearray.h>

#include <QtCore/qmetatype.h>

#if !defined(Q\_MOC\_OUTPUT\_REVISION)

#error "The header file 'start\_menu.h' doesn't include <QObject>."

#elif Q\_MOC\_OUTPUT\_REVISION != 67

#error "This file was generated using the moc from 5.10.1. It"

#error "cannot be used with the include files from this version of Qt."

#error "(The moc has changed too much.)"

#endif

QT\_BEGIN\_MOC\_NAMESPACE

QT\_WARNING\_PUSH

QT\_WARNING\_DISABLE\_DEPRECATED

struct qt\_meta\_stringdata\_start\_menu\_t {

QByteArrayData data[4];

char stringdata0[76];

};

#define QT\_MOC\_LITERAL(idx, ofs, len) \

Q\_STATIC\_BYTE\_ARRAY\_DATA\_HEADER\_INITIALIZER\_WITH\_OFFSET(len, \

qptrdiff(offsetof(qt\_meta\_stringdata\_start\_menu\_t, stringdata0) + ofs \

- idx \* sizeof(QByteArrayData)) \

)

static const qt\_meta\_stringdata\_start\_menu\_t qt\_meta\_stringdata\_start\_menu = {

{

QT\_MOC\_LITERAL(0, 0, 10), // "start\_menu"

QT\_MOC\_LITERAL(1, 11, 28), // "on\_create\_new\_wallet\_clicked"

QT\_MOC\_LITERAL(2, 40, 0), // ""

QT\_MOC\_LITERAL(3, 41, 34) // "on\_restore\_existing\_wallet\_cl..."

},

"start\_menu\0on\_create\_new\_wallet\_clicked\0"

"\0on\_restore\_existing\_wallet\_clicked"

};

#undef QT\_MOC\_LITERAL

static const uint qt\_meta\_data\_start\_menu[] = {

// content:

7, // revision

0, // classname

0, 0, // classinfo

2, 14, // methods

0, 0, // properties

0, 0, // enums/sets

0, 0, // constructors

0, // flags

0, // signalCount

// slots: name, argc, parameters, tag, flags

1, 0, 24, 2, 0x08 /\* Private \*/,

3, 0, 25, 2, 0x08 /\* Private \*/,

// slots: parameters

QMetaType::Void,

QMetaType::Void,

0 // eod

};

void start\_menu::qt\_static\_metacall(QObject \*\_o, QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

if (\_c == QMetaObject::InvokeMetaMethod) {

start\_menu \*\_t = static\_cast<start\_menu \*>(\_o);

Q\_UNUSED(\_t)

switch (\_id) {

case 0: \_t->on\_create\_new\_wallet\_clicked(); break;

case 1: \_t->on\_restore\_existing\_wallet\_clicked(); break;

default: ;

}

}

Q\_UNUSED(\_a);

}

QT\_INIT\_METAOBJECT const QMetaObject start\_menu::staticMetaObject = {

{ &QDialog::staticMetaObject, qt\_meta\_stringdata\_start\_menu.data,

qt\_meta\_data\_start\_menu, qt\_static\_metacall, nullptr, nullptr}

};

const QMetaObject \*start\_menu::metaObject() const

{

return QObject::d\_ptr->metaObject ? QObject::d\_ptr->dynamicMetaObject() : &staticMetaObject;

}

void \*start\_menu::qt\_metacast(const char \*\_clname)

{

if (!\_clname) return nullptr;

if (!strcmp(\_clname, qt\_meta\_stringdata\_start\_menu.stringdata0))

return static\_cast<void\*>(this);

return QDialog::qt\_metacast(\_clname);

}

int start\_menu::qt\_metacall(QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

\_id = QDialog::qt\_metacall(\_c, \_id, \_a);

if (\_id < 0)

return \_id;

if (\_c == QMetaObject::InvokeMetaMethod) {

if (\_id < 2)

qt\_static\_metacall(this, \_c, \_id, \_a);

\_id -= 2;

} else if (\_c == QMetaObject::RegisterMethodArgumentMetaType) {

if (\_id < 2)

\*reinterpret\_cast<int\*>(\_a[0]) = -1;

\_id -= 2;

}

return \_id;

}

QT\_WARNING\_POP

QT\_END\_MOC\_NAMESPACE

~~~~~~~~~~~~~~~~~~~~~Source code for file ui\_app.h~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'app.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.10.1

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef UI\_APP\_H

#define UI\_APP\_H

#include <QtCore/QVariant>

#include <QtWidgets/QAction>

#include <QtWidgets/QApplication>

#include <QtWidgets/QButtonGroup>

#include <QtWidgets/QHeaderView>

#include <QtWidgets/QLabel>

#include <QtWidgets/QMainWindow>

#include <QtWidgets/QMenuBar>

#include <QtWidgets/QStatusBar>

#include <QtWidgets/QToolBar>

#include <QtWidgets/QWidget>

QT\_BEGIN\_NAMESPACE

class Ui\_app

{

public:

QWidget \*centralWidget;

QLabel \*debuggerLabel;

QMenuBar \*menuBar;

QToolBar \*mainToolBar;

QStatusBar \*statusBar;

void setupUi(QMainWindow \*app)

{

if (app->objectName().isEmpty())

app->setObjectName(QStringLiteral("app"));

app->resize(953, 467);

centralWidget = new QWidget(app);

centralWidget->setObjectName(QStringLiteral("centralWidget"));

debuggerLabel = new QLabel(centralWidget);

debuggerLabel->setObjectName(QStringLiteral("debuggerLabel"));

debuggerLabel->setGeometry(QRect(160, 180, 60, 16));

app->setCentralWidget(centralWidget);

menuBar = new QMenuBar(app);

menuBar->setObjectName(QStringLiteral("menuBar"));

menuBar->setGeometry(QRect(0, 0, 953, 22));

app->setMenuBar(menuBar);

mainToolBar = new QToolBar(app);

mainToolBar->setObjectName(QStringLiteral("mainToolBar"));

app->addToolBar(Qt::TopToolBarArea, mainToolBar);

statusBar = new QStatusBar(app);

statusBar->setObjectName(QStringLiteral("statusBar"));

app->setStatusBar(statusBar);

retranslateUi(app);

QMetaObject::connectSlotsByName(app);

} // setupUi

void retranslateUi(QMainWindow \*app)

{

app->setWindowTitle(QApplication::translate("app", "app", nullptr));

debuggerLabel->setText(QApplication::translate("app", "TextLabel", nullptr));

} // retranslateUi

};

namespace Ui {

class app: public Ui\_app {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // UI\_APP\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file moc\_predefs.h~~~~~~~~~~~~~~~~~~~~~

#define OBJC\_NEW\_PROPERTIES 1

#define \_LP64 1

#define \_\_APPLE\_CC\_\_ 6000

#define \_\_APPLE\_\_ 1

#define \_\_ATOMIC\_ACQUIRE 2

#define \_\_ATOMIC\_ACQ\_REL 4

#define \_\_ATOMIC\_CONSUME 1

#define \_\_ATOMIC\_RELAXED 0

#define \_\_ATOMIC\_RELEASE 3

#define \_\_ATOMIC\_SEQ\_CST 5

#define \_\_BIGGEST\_ALIGNMENT\_\_ 16

#define \_\_BLOCKS\_\_ 1

#define \_\_BYTE\_ORDER\_\_ \_\_ORDER\_LITTLE\_ENDIAN\_\_

#define \_\_CHAR16\_TYPE\_\_ unsigned short

#define \_\_CHAR32\_TYPE\_\_ unsigned int

#define \_\_CHAR\_BIT\_\_ 8

#define \_\_CONSTANT\_CFSTRINGS\_\_ 1

#define \_\_DBL\_DECIMAL\_DIG\_\_ 17

#define \_\_DBL\_DENORM\_MIN\_\_ 4.9406564584124654e-324

#define \_\_DBL\_DIG\_\_ 15

#define \_\_DBL\_EPSILON\_\_ 2.2204460492503131e-16

#define \_\_DBL\_HAS\_DENORM\_\_ 1

#define \_\_DBL\_HAS\_INFINITY\_\_ 1

#define \_\_DBL\_HAS\_QUIET\_NAN\_\_ 1

#define \_\_DBL\_MANT\_DIG\_\_ 53

#define \_\_DBL\_MAX\_10\_EXP\_\_ 308

#define \_\_DBL\_MAX\_EXP\_\_ 1024

#define \_\_DBL\_MAX\_\_ 1.7976931348623157e+308

#define \_\_DBL\_MIN\_10\_EXP\_\_ (-307)

#define \_\_DBL\_MIN\_EXP\_\_ (-1021)

#define \_\_DBL\_MIN\_\_ 2.2250738585072014e-308

#define \_\_DECIMAL\_DIG\_\_ \_\_LDBL\_DECIMAL\_DIG\_\_

#define \_\_DEPRECATED 1

#define \_\_DYNAMIC\_\_ 1

#define \_\_ENVIRONMENT\_MAC\_OS\_X\_VERSION\_MIN\_REQUIRED\_\_ 101000

#define \_\_EXCEPTIONS 1

#define \_\_FINITE\_MATH\_ONLY\_\_ 0

#define \_\_FLT\_DECIMAL\_DIG\_\_ 9

#define \_\_FLT\_DENORM\_MIN\_\_ 1.40129846e-45F

#define \_\_FLT\_DIG\_\_ 6

#define \_\_FLT\_EPSILON\_\_ 1.19209290e-7F

#define \_\_FLT\_EVAL\_METHOD\_\_ 0

#define \_\_FLT\_HAS\_DENORM\_\_ 1

#define \_\_FLT\_HAS\_INFINITY\_\_ 1

#define \_\_FLT\_HAS\_QUIET\_NAN\_\_ 1

#define \_\_FLT\_MANT\_DIG\_\_ 24

#define \_\_FLT\_MAX\_10\_EXP\_\_ 38

#define \_\_FLT\_MAX\_EXP\_\_ 128

#define \_\_FLT\_MAX\_\_ 3.40282347e+38F

#define \_\_FLT\_MIN\_10\_EXP\_\_ (-37)

#define \_\_FLT\_MIN\_EXP\_\_ (-125)

#define \_\_FLT\_MIN\_\_ 1.17549435e-38F

#define \_\_FLT\_RADIX\_\_ 2

#define \_\_FXSR\_\_ 1

#define \_\_GCC\_ATOMIC\_BOOL\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_CHAR16\_T\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_CHAR32\_T\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_CHAR\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_INT\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_LLONG\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_LONG\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_POINTER\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_SHORT\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_TEST\_AND\_SET\_TRUEVAL 1

#define \_\_GCC\_ATOMIC\_WCHAR\_T\_LOCK\_FREE 2

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_1 1

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_16 1

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_2 1

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_4 1

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_8 1

#define \_\_GLIBCXX\_BITSIZE\_INT\_N\_0 128

#define \_\_GLIBCXX\_TYPE\_INT\_N\_0 \_\_int128

#define \_\_GNUC\_GNU\_INLINE\_\_ 1

#define \_\_GNUC\_MINOR\_\_ 2

#define \_\_GNUC\_PATCHLEVEL\_\_ 1

#define \_\_GNUC\_\_ 4

#define \_\_GNUG\_\_ 4

#define \_\_GXX\_ABI\_VERSION 1002

#define \_\_GXX\_EXPERIMENTAL\_CXX0X\_\_ 1

#define \_\_GXX\_RTTI 1

#define \_\_GXX\_WEAK\_\_ 1

#define \_\_INT16\_C\_SUFFIX\_\_

#define \_\_INT16\_FMTd\_\_ "hd"

#define \_\_INT16\_FMTi\_\_ "hi"

#define \_\_INT16\_MAX\_\_ 32767

#define \_\_INT16\_TYPE\_\_ short

#define \_\_INT32\_C\_SUFFIX\_\_

#define \_\_INT32\_FMTd\_\_ "d"

#define \_\_INT32\_FMTi\_\_ "i"

#define \_\_INT32\_MAX\_\_ 2147483647

#define \_\_INT32\_TYPE\_\_ int

#define \_\_INT64\_C\_SUFFIX\_\_ LL

#define \_\_INT64\_FMTd\_\_ "lld"

#define \_\_INT64\_FMTi\_\_ "lli"

#define \_\_INT64\_MAX\_\_ 9223372036854775807LL

#define \_\_INT64\_TYPE\_\_ long long int

#define \_\_INT8\_C\_SUFFIX\_\_

#define \_\_INT8\_FMTd\_\_ "hhd"

#define \_\_INT8\_FMTi\_\_ "hhi"

#define \_\_INT8\_MAX\_\_ 127

#define \_\_INT8\_TYPE\_\_ signed char

#define \_\_INTMAX\_C\_SUFFIX\_\_ L

#define \_\_INTMAX\_FMTd\_\_ "ld"

#define \_\_INTMAX\_FMTi\_\_ "li"

#define \_\_INTMAX\_MAX\_\_ 9223372036854775807L

#define \_\_INTMAX\_TYPE\_\_ long int

#define \_\_INTMAX\_WIDTH\_\_ 64

#define \_\_INTPTR\_FMTd\_\_ "ld"

#define \_\_INTPTR\_FMTi\_\_ "li"

#define \_\_INTPTR\_MAX\_\_ 9223372036854775807L

#define \_\_INTPTR\_TYPE\_\_ long int

#define \_\_INTPTR\_WIDTH\_\_ 64

#define \_\_INT\_FAST16\_FMTd\_\_ "hd"

#define \_\_INT\_FAST16\_FMTi\_\_ "hi"

#define \_\_INT\_FAST16\_MAX\_\_ 32767

#define \_\_INT\_FAST16\_TYPE\_\_ short

#define \_\_INT\_FAST32\_FMTd\_\_ "d"

#define \_\_INT\_FAST32\_FMTi\_\_ "i"

#define \_\_INT\_FAST32\_MAX\_\_ 2147483647

#define \_\_INT\_FAST32\_TYPE\_\_ int

#define \_\_INT\_FAST64\_FMTd\_\_ "ld"

#define \_\_INT\_FAST64\_FMTi\_\_ "li"

#define \_\_INT\_FAST64\_MAX\_\_ 9223372036854775807L

#define \_\_INT\_FAST64\_TYPE\_\_ long int

#define \_\_INT\_FAST8\_FMTd\_\_ "hhd"

#define \_\_INT\_FAST8\_FMTi\_\_ "hhi"

#define \_\_INT\_FAST8\_MAX\_\_ 127

#define \_\_INT\_FAST8\_TYPE\_\_ signed char

#define \_\_INT\_LEAST16\_FMTd\_\_ "hd"

#define \_\_INT\_LEAST16\_FMTi\_\_ "hi"

#define \_\_INT\_LEAST16\_MAX\_\_ 32767

#define \_\_INT\_LEAST16\_TYPE\_\_ short

#define \_\_INT\_LEAST32\_FMTd\_\_ "d"

#define \_\_INT\_LEAST32\_FMTi\_\_ "i"

#define \_\_INT\_LEAST32\_MAX\_\_ 2147483647

#define \_\_INT\_LEAST32\_TYPE\_\_ int

#define \_\_INT\_LEAST64\_FMTd\_\_ "ld"

#define \_\_INT\_LEAST64\_FMTi\_\_ "li"

#define \_\_INT\_LEAST64\_MAX\_\_ 9223372036854775807L

#define \_\_INT\_LEAST64\_TYPE\_\_ long int

#define \_\_INT\_LEAST8\_FMTd\_\_ "hhd"

#define \_\_INT\_LEAST8\_FMTi\_\_ "hhi"

#define \_\_INT\_LEAST8\_MAX\_\_ 127

#define \_\_INT\_LEAST8\_TYPE\_\_ signed char

#define \_\_INT\_MAX\_\_ 2147483647

#define \_\_LDBL\_DECIMAL\_DIG\_\_ 21

#define \_\_LDBL\_DENORM\_MIN\_\_ 3.64519953188247460253e-4951L

#define \_\_LDBL\_DIG\_\_ 18

#define \_\_LDBL\_EPSILON\_\_ 1.08420217248550443401e-19L

#define \_\_LDBL\_HAS\_DENORM\_\_ 1

#define \_\_LDBL\_HAS\_INFINITY\_\_ 1

#define \_\_LDBL\_HAS\_QUIET\_NAN\_\_ 1

#define \_\_LDBL\_MANT\_DIG\_\_ 64

#define \_\_LDBL\_MAX\_10\_EXP\_\_ 4932

#define \_\_LDBL\_MAX\_EXP\_\_ 16384

#define \_\_LDBL\_MAX\_\_ 1.18973149535723176502e+4932L

#define \_\_LDBL\_MIN\_10\_EXP\_\_ (-4931)

#define \_\_LDBL\_MIN\_EXP\_\_ (-16381)

#define \_\_LDBL\_MIN\_\_ 3.36210314311209350626e-4932L

#define \_\_LITTLE\_ENDIAN\_\_ 1

#define \_\_LONG\_LONG\_MAX\_\_ 9223372036854775807LL

#define \_\_LONG\_MAX\_\_ 9223372036854775807L

#define \_\_LP64\_\_ 1

#define \_\_MACH\_\_ 1

#define \_\_MMX\_\_ 1

#define \_\_NO\_INLINE\_\_ 1

#define \_\_NO\_MATH\_INLINES 1

#define \_\_OBJC\_BOOL\_IS\_BOOL 0

#define \_\_ORDER\_BIG\_ENDIAN\_\_ 4321

#define \_\_ORDER\_LITTLE\_ENDIAN\_\_ 1234

#define \_\_ORDER\_PDP\_ENDIAN\_\_ 3412

#define \_\_PIC\_\_ 2

#define \_\_POINTER\_WIDTH\_\_ 64

#define \_\_PRAGMA\_REDEFINE\_EXTNAME 1

#define \_\_PTRDIFF\_FMTd\_\_ "ld"

#define \_\_PTRDIFF\_FMTi\_\_ "li"

#define \_\_PTRDIFF\_MAX\_\_ 9223372036854775807L

#define \_\_PTRDIFF\_TYPE\_\_ long int

#define \_\_PTRDIFF\_WIDTH\_\_ 64

#define \_\_REGISTER\_PREFIX\_\_

#define \_\_SCHAR\_MAX\_\_ 127

#define \_\_SHRT\_MAX\_\_ 32767

#define \_\_SIG\_ATOMIC\_MAX\_\_ 2147483647

#define \_\_SIG\_ATOMIC\_WIDTH\_\_ 32

#define \_\_SIZEOF\_DOUBLE\_\_ 8

#define \_\_SIZEOF\_FLOAT\_\_ 4

#define \_\_SIZEOF\_INT128\_\_ 16

#define \_\_SIZEOF\_INT\_\_ 4

#define \_\_SIZEOF\_LONG\_DOUBLE\_\_ 16

#define \_\_SIZEOF\_LONG\_LONG\_\_ 8

#define \_\_SIZEOF\_LONG\_\_ 8

#define \_\_SIZEOF\_POINTER\_\_ 8

#define \_\_SIZEOF\_PTRDIFF\_T\_\_ 8

#define \_\_SIZEOF\_SHORT\_\_ 2

#define \_\_SIZEOF\_SIZE\_T\_\_ 8

#define \_\_SIZEOF\_WCHAR\_T\_\_ 4

#define \_\_SIZEOF\_WINT\_T\_\_ 4

#define \_\_SIZE\_FMTX\_\_ "lX"

#define \_\_SIZE\_FMTo\_\_ "lo"

#define \_\_SIZE\_FMTu\_\_ "lu"

#define \_\_SIZE\_FMTx\_\_ "lx"

#define \_\_SIZE\_MAX\_\_ 18446744073709551615UL

#define \_\_SIZE\_TYPE\_\_ long unsigned int

#define \_\_SIZE\_WIDTH\_\_ 64

#define \_\_SSE2\_MATH\_\_ 1

#define \_\_SSE2\_\_ 1

#define \_\_SSE3\_\_ 1

#define \_\_SSE\_MATH\_\_ 1

#define \_\_SSE\_\_ 1

#define \_\_SSP\_\_ 1

#define \_\_SSSE3\_\_ 1

#define \_\_STDCPP\_DEFAULT\_NEW\_ALIGNMENT\_\_ 16UL

#define \_\_STDC\_HOSTED\_\_ 1

#define \_\_STDC\_NO\_THREADS\_\_ 1

#define \_\_STDC\_UTF\_16\_\_ 1

#define \_\_STDC\_UTF\_32\_\_ 1

#define \_\_STDC\_\_ 1

#define \_\_UINT16\_C\_SUFFIX\_\_

#define \_\_UINT16\_FMTX\_\_ "hX"

#define \_\_UINT16\_FMTo\_\_ "ho"

#define \_\_UINT16\_FMTu\_\_ "hu"

#define \_\_UINT16\_FMTx\_\_ "hx"

#define \_\_UINT16\_MAX\_\_ 65535

#define \_\_UINT16\_TYPE\_\_ unsigned short

#define \_\_UINT32\_C\_SUFFIX\_\_ U

#define \_\_UINT32\_FMTX\_\_ "X"

#define \_\_UINT32\_FMTo\_\_ "o"

#define \_\_UINT32\_FMTu\_\_ "u"

#define \_\_UINT32\_FMTx\_\_ "x"

#define \_\_UINT32\_MAX\_\_ 4294967295U

#define \_\_UINT32\_TYPE\_\_ unsigned int

#define \_\_UINT64\_C\_SUFFIX\_\_ ULL

#define \_\_UINT64\_FMTX\_\_ "llX"

#define \_\_UINT64\_FMTo\_\_ "llo"

#define \_\_UINT64\_FMTu\_\_ "llu"

#define \_\_UINT64\_FMTx\_\_ "llx"

#define \_\_UINT64\_MAX\_\_ 18446744073709551615ULL

#define \_\_UINT64\_TYPE\_\_ long long unsigned int

#define \_\_UINT8\_C\_SUFFIX\_\_

#define \_\_UINT8\_FMTX\_\_ "hhX"

#define \_\_UINT8\_FMTo\_\_ "hho"

#define \_\_UINT8\_FMTu\_\_ "hhu"

#define \_\_UINT8\_FMTx\_\_ "hhx"

#define \_\_UINT8\_MAX\_\_ 255

#define \_\_UINT8\_TYPE\_\_ unsigned char

#define \_\_UINTMAX\_C\_SUFFIX\_\_ UL

#define \_\_UINTMAX\_FMTX\_\_ "lX"

#define \_\_UINTMAX\_FMTo\_\_ "lo"

#define \_\_UINTMAX\_FMTu\_\_ "lu"

#define \_\_UINTMAX\_FMTx\_\_ "lx"

#define \_\_UINTMAX\_MAX\_\_ 18446744073709551615UL

#define \_\_UINTMAX\_TYPE\_\_ long unsigned int

#define \_\_UINTMAX\_WIDTH\_\_ 64

#define \_\_UINTPTR\_FMTX\_\_ "lX"

#define \_\_UINTPTR\_FMTo\_\_ "lo"

#define \_\_UINTPTR\_FMTu\_\_ "lu"

#define \_\_UINTPTR\_FMTx\_\_ "lx"

#define \_\_UINTPTR\_MAX\_\_ 18446744073709551615UL

#define \_\_UINTPTR\_TYPE\_\_ long unsigned int

#define \_\_UINTPTR\_WIDTH\_\_ 64

#define \_\_UINT\_FAST16\_FMTX\_\_ "hX"

#define \_\_UINT\_FAST16\_FMTo\_\_ "ho"

#define \_\_UINT\_FAST16\_FMTu\_\_ "hu"

#define \_\_UINT\_FAST16\_FMTx\_\_ "hx"

#define \_\_UINT\_FAST16\_MAX\_\_ 65535

#define \_\_UINT\_FAST16\_TYPE\_\_ unsigned short

#define \_\_UINT\_FAST32\_FMTX\_\_ "X"

#define \_\_UINT\_FAST32\_FMTo\_\_ "o"

#define \_\_UINT\_FAST32\_FMTu\_\_ "u"

#define \_\_UINT\_FAST32\_FMTx\_\_ "x"

#define \_\_UINT\_FAST32\_MAX\_\_ 4294967295U

#define \_\_UINT\_FAST32\_TYPE\_\_ unsigned int

#define \_\_UINT\_FAST64\_FMTX\_\_ "lX"

#define \_\_UINT\_FAST64\_FMTo\_\_ "lo"

#define \_\_UINT\_FAST64\_FMTu\_\_ "lu"

#define \_\_UINT\_FAST64\_FMTx\_\_ "lx"

#define \_\_UINT\_FAST64\_MAX\_\_ 18446744073709551615UL

#define \_\_UINT\_FAST64\_TYPE\_\_ long unsigned int

#define \_\_UINT\_FAST8\_FMTX\_\_ "hhX"

#define \_\_UINT\_FAST8\_FMTo\_\_ "hho"

#define \_\_UINT\_FAST8\_FMTu\_\_ "hhu"

#define \_\_UINT\_FAST8\_FMTx\_\_ "hhx"

#define \_\_UINT\_FAST8\_MAX\_\_ 255

#define \_\_UINT\_FAST8\_TYPE\_\_ unsigned char

#define \_\_UINT\_LEAST16\_FMTX\_\_ "hX"

#define \_\_UINT\_LEAST16\_FMTo\_\_ "ho"

#define \_\_UINT\_LEAST16\_FMTu\_\_ "hu"

#define \_\_UINT\_LEAST16\_FMTx\_\_ "hx"

#define \_\_UINT\_LEAST16\_MAX\_\_ 65535

#define \_\_UINT\_LEAST16\_TYPE\_\_ unsigned short

#define \_\_UINT\_LEAST32\_FMTX\_\_ "X"

#define \_\_UINT\_LEAST32\_FMTo\_\_ "o"

#define \_\_UINT\_LEAST32\_FMTu\_\_ "u"

#define \_\_UINT\_LEAST32\_FMTx\_\_ "x"

#define \_\_UINT\_LEAST32\_MAX\_\_ 4294967295U

#define \_\_UINT\_LEAST32\_TYPE\_\_ unsigned int

#define \_\_UINT\_LEAST64\_FMTX\_\_ "lX"

#define \_\_UINT\_LEAST64\_FMTo\_\_ "lo"

#define \_\_UINT\_LEAST64\_FMTu\_\_ "lu"

#define \_\_UINT\_LEAST64\_FMTx\_\_ "lx"

#define \_\_UINT\_LEAST64\_MAX\_\_ 18446744073709551615UL

#define \_\_UINT\_LEAST64\_TYPE\_\_ long unsigned int

#define \_\_UINT\_LEAST8\_FMTX\_\_ "hhX"

#define \_\_UINT\_LEAST8\_FMTo\_\_ "hho"

#define \_\_UINT\_LEAST8\_FMTu\_\_ "hhu"

#define \_\_UINT\_LEAST8\_FMTx\_\_ "hhx"

#define \_\_UINT\_LEAST8\_MAX\_\_ 255

#define \_\_UINT\_LEAST8\_TYPE\_\_ unsigned char

#define \_\_USER\_LABEL\_PREFIX\_\_ \_

#define \_\_VERSION\_\_ "4.2.1 Compatible Apple LLVM 9.0.0 (clang-900.0.39.2)"

#define \_\_WCHAR\_MAX\_\_ 2147483647

#define \_\_WCHAR\_TYPE\_\_ int

#define \_\_WCHAR\_WIDTH\_\_ 32

#define \_\_WINT\_TYPE\_\_ int

#define \_\_WINT\_WIDTH\_\_ 32

#define \_\_amd64 1

#define \_\_amd64\_\_ 1

#define \_\_apple\_build\_version\_\_ 9000039

#define \_\_block \_\_attribute\_\_((\_\_blocks\_\_(byref)))

#define \_\_clang\_\_ 1

#define \_\_clang\_major\_\_ 9

#define \_\_clang\_minor\_\_ 0

#define \_\_clang\_patchlevel\_\_ 0

#define \_\_clang\_version\_\_ "9.0.0 (clang-900.0.39.2)"

#define \_\_core2 1

#define \_\_core2\_\_ 1

#define \_\_cplusplus 201103L

#define \_\_cpp\_alias\_templates 200704

#define \_\_cpp\_attributes 200809

#define \_\_cpp\_constexpr 200704

#define \_\_cpp\_decltype 200707

#define \_\_cpp\_delegating\_constructors 200604

#define \_\_cpp\_exceptions 199711

#define \_\_cpp\_inheriting\_constructors 201511

#define \_\_cpp\_initializer\_lists 200806

#define \_\_cpp\_lambdas 200907

#define \_\_cpp\_nsdmi 200809

#define \_\_cpp\_range\_based\_for 200907

#define \_\_cpp\_raw\_strings 200710

#define \_\_cpp\_ref\_qualifiers 200710

#define \_\_cpp\_rtti 199711

#define \_\_cpp\_rvalue\_references 200610

#define \_\_cpp\_static\_assert 200410

#define \_\_cpp\_unicode\_characters 200704

#define \_\_cpp\_unicode\_literals 200710

#define \_\_cpp\_user\_defined\_literals 200809

#define \_\_cpp\_variadic\_templates 200704

#define \_\_llvm\_\_ 1

#define \_\_nonnull \_Nonnull

#define \_\_null\_unspecified \_Null\_unspecified

#define \_\_nullable \_Nullable

#define \_\_pic\_\_ 2

#define \_\_private\_extern\_\_ extern

#define \_\_strong

#define \_\_tune\_core2\_\_ 1

#define \_\_unsafe\_unretained

#define \_\_weak \_\_attribute\_\_((objc\_gc(weak)))

#define \_\_x86\_64 1

#define \_\_x86\_64\_\_ 1

~~~~~~~~~~~~~~~~~~~~~Source code for file moc\_restore\_wallet.cpp~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Meta object code from reading C++ file 'restore\_wallet.h'

\*\*

\*\* Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)

\*\*

\*\* WARNING! All changes made in this file will be lost!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "../Atlas/restore\_wallet.h"

#include <QtCore/qbytearray.h>

#include <QtCore/qmetatype.h>

#if !defined(Q\_MOC\_OUTPUT\_REVISION)

#error "The header file 'restore\_wallet.h' doesn't include <QObject>."

#elif Q\_MOC\_OUTPUT\_REVISION != 67

#error "This file was generated using the moc from 5.10.1. It"

#error "cannot be used with the include files from this version of Qt."

#error "(The moc has changed too much.)"

#endif

QT\_BEGIN\_MOC\_NAMESPACE

QT\_WARNING\_PUSH

QT\_WARNING\_DISABLE\_DEPRECATED

struct qt\_meta\_stringdata\_restore\_wallet\_t {

QByteArrayData data[1];

char stringdata0[15];

};

#define QT\_MOC\_LITERAL(idx, ofs, len) \

Q\_STATIC\_BYTE\_ARRAY\_DATA\_HEADER\_INITIALIZER\_WITH\_OFFSET(len, \

qptrdiff(offsetof(qt\_meta\_stringdata\_restore\_wallet\_t, stringdata0) + ofs \

- idx \* sizeof(QByteArrayData)) \

)

static const qt\_meta\_stringdata\_restore\_wallet\_t qt\_meta\_stringdata\_restore\_wallet = {

{

QT\_MOC\_LITERAL(0, 0, 14) // "restore\_wallet"

},

"restore\_wallet"

};

#undef QT\_MOC\_LITERAL

static const uint qt\_meta\_data\_restore\_wallet[] = {

// content:

7, // revision

0, // classname

0, 0, // classinfo

0, 0, // methods

0, 0, // properties

0, 0, // enums/sets

0, 0, // constructors

0, // flags

0, // signalCount

0 // eod

};

void restore\_wallet::qt\_static\_metacall(QObject \*\_o, QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

Q\_UNUSED(\_o);

Q\_UNUSED(\_id);

Q\_UNUSED(\_c);

Q\_UNUSED(\_a);

}

QT\_INIT\_METAOBJECT const QMetaObject restore\_wallet::staticMetaObject = {

{ &QDialog::staticMetaObject, qt\_meta\_stringdata\_restore\_wallet.data,

qt\_meta\_data\_restore\_wallet, qt\_static\_metacall, nullptr, nullptr}

};

const QMetaObject \*restore\_wallet::metaObject() const

{

return QObject::d\_ptr->metaObject ? QObject::d\_ptr->dynamicMetaObject() : &staticMetaObject;

}

void \*restore\_wallet::qt\_metacast(const char \*\_clname)

{

if (!\_clname) return nullptr;

if (!strcmp(\_clname, qt\_meta\_stringdata\_restore\_wallet.stringdata0))

return static\_cast<void\*>(this);

return QDialog::qt\_metacast(\_clname);

}

int restore\_wallet::qt\_metacall(QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

\_id = QDialog::qt\_metacall(\_c, \_id, \_a);

return \_id;

}

QT\_WARNING\_POP

QT\_END\_MOC\_NAMESPACE

~~~~~~~~~~~~~~~~~~~~~Source code for file restore\_wallet.cpp~~~~~~~~~~~~~~~~~~~~~

#include "restore\_wallet.h"

#include "ui\_restore\_wallet.h"

#include <string>

restore\_wallet::restore\_wallet(QWidget \*parent) :

QDialog(parent),

ui(new Ui::restore\_wallet)

{

ui->setupUi(this);

}

restore\_wallet::~restore\_wallet()

{

delete ui;

}

const std::vector<std::string> restore\_wallet::get\_word\_list()

{

// Vector holds the word list of the mnemomic 12-word phrase.

std::vector<std::string> word\_list;

// Append each word to vector.

word\_list.push\_back(ui->word\_1->text().toStdString());

word\_list.push\_back(ui->word\_2->text().toStdString());

word\_list.push\_back(ui->word\_3->text().toStdString());

word\_list.push\_back(ui->word\_4->text().toStdString());

word\_list.push\_back(ui->word\_5->text().toStdString());

word\_list.push\_back(ui->word\_6->text().toStdString());

word\_list.push\_back(ui->word\_7->text().toStdString());

word\_list.push\_back(ui->word\_8->text().toStdString());

word\_list.push\_back(ui->word\_9->text().toStdString());

word\_list.push\_back(ui->word\_10->text().toStdString());

word\_list.push\_back(ui->word\_11->text().toStdString());

word\_list.push\_back(ui->word\_12->text().toStdString());

return word\_list;

}

~~~~~~~~~~~~~~~~~~~~~Source code for file app.h~~~~~~~~~~~~~~~~~~~~~

#ifndef APP\_H

#define APP\_H

#include <QErrorMessage>

#include <QMainWindow>

#include "restore\_wallet.h"

#include "start\_menu.h"

#include "../../wallet/stdafx.h"

namespace Ui {

class app;

}

class app : public QMainWindow

{

Q\_OBJECT

public:

explicit app(QWidget \*parent = 0);

~app();

private slots:

// Tab is changed on main application menu.

void on\_tabWidget\_tabBarClicked(int index);

// Copy bitcoin address is clicked.

void on\_copy\_btc\_address\_clicked();

// Fee slider is moved on send transaction tab.

void on\_fee\_slider\_sliderMoved(int position);

// Send transaction button is clicked on send transaction tab.

void on\_send\_tx\_clicked();

void on\_run\_script\_btn\_clicked();

private:

// Error Dialog

QErrorMessage error\_msg;

// Wallet objects.

Wallet \* wallet;

Network \* network;

Script \* script;

// New or Restore Wallet.

std::string menu\_choice;

Ui::app \*ui;

restore\_wallet \* restore\_wallet;

start\_menu \* start\_menu;

QLayout \* script\_layout;

// Mnemonic word list.

std::vector<std::string> word\_list;

// Initialize wallet.

void init\_start\_menu();

void init\_wallet();

void get\_mnemonic\_phrase();

// Change widgets on main tab.

void set\_main\_tab();

void set\_available\_payment\_address();

void set\_btc\_recieved();

void set\_btc\_sent();

void set\_btc\_balance();

// Change widgets on send tab.

void set\_send\_tab();

bool send\_transaction();

// Change widgets on history tab.

void set\_history\_tab();

// Input validation for send tab.

bool is\_validate\_tx();

bool is\_valid\_address();

// Change widgets on analytics tab.

void set\_analytics\_tab();

// Change widgets on script tab.

void set\_script\_tab();

void run\_script();

void write\_to\_script\_console(std::string msg);

};

#endif // APP\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file start\_menu.cpp~~~~~~~~~~~~~~~~~~~~~

#include "start\_menu.h"

#include "ui\_start\_menu.h"

start\_menu::start\_menu(QWidget \*parent) :

QDialog(parent),

ui(new Ui::start\_menu)

{

ui->setupUi(this);

}

start\_menu::~start\_menu()

{

delete ui;

}

std::string start\_menu::get\_menu\_choice() const

{

return menu\_choice;

}

void start\_menu::on\_create\_new\_wallet\_clicked()

{

menu\_choice = NEW\_WALLET;

this->close();

}

void start\_menu::on\_restore\_existing\_wallet\_clicked()

{

menu\_choice = RESTORE\_WALLET;

this->close();

}

~~~~~~~~~~~~~~~~~~~~~Source code for file restore\_wallet.h~~~~~~~~~~~~~~~~~~~~~

#ifndef RESTORE\_WALLET\_H

#define RESTORE\_WALLET\_H

#include <QDialog>

#include <string>

namespace Ui {

class restore\_wallet;

}

class restore\_wallet : public QDialog

{

Q\_OBJECT

public:

explicit restore\_wallet(QWidget \*parent = 0);

~restore\_wallet();

// Returns a vector of the 12-word phrase.

const std::vector<std::string> get\_word\_list();

// TODO:

// QValidator, setValidator

// When value in lineedit is changed, signal is emitted, check if the value change is valid.

// Require all lines to be filled out.

private:

Ui::restore\_wallet \*ui;

};

#endif // RESTORE\_WALLET\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file start\_menu.h~~~~~~~~~~~~~~~~~~~~~

#ifndef START\_MENU\_H

#define START\_MENU\_H

#include <QDialog>

namespace Ui {

class start\_menu;

}

class start\_menu : public QDialog

{

Q\_OBJECT

public:

explicit start\_menu(QWidget \*parent = 0);

~start\_menu();

std::string get\_menu\_choice() const;

private slots:

// New wallet will be made.

void on\_create\_new\_wallet\_clicked();

// Restore wallet will be made.s

void on\_restore\_existing\_wallet\_clicked();

private:

const std::string NEW\_WALLET = "new";

const std::string RESTORE\_WALLET = "restore";

std::string menu\_choice;

Ui::start\_menu \*ui;

};

#endif // START\_MENU\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file main.cpp~~~~~~~~~~~~~~~~~~~~~

#include "app.h"

#include "restore\_wallet.h"

#include <QApplication>

#include "start\_menu.h"

#include "../../wallet/stdafx.h"

int main(int argc, char \*argv[])

{

// Launch main application window.

QApplication a(argc, argv);

app w;

w.show();

return a.exec();

}

~~~~~~~~~~~~~~~~~~~~~Source code for file app.cpp~~~~~~~~~~~~~~~~~~~~~

#include "app.h"

#include "ui\_app.h"

#include "restore\_wallet.h"

#include <string>

#include <vector>

#include <QDebug>

#include "../../wallet/stdafx.h"

#include <QClipboard>

#include <QRadioButton>

#include <qboxlayout.h>

/\*\*

\* @brief Constructor for app::app

\* @param parent

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

app::app(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::app)

{

ui->setupUi(this);

// Starts immediate dialog box asking user for new/restore wallet.

init\_start\_menu();

init\_wallet();

set\_main\_tab();

// Initialize network.

network = new Network();

// Initialize script.

script = new Script();

}

/\*\*

\* @brief Desturctor for app::~app

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

app::~app()

{

delete ui;

}

/\*\*

\* @brief Asks user if they would like to start a new wallet or restore wallet. Creates new dialog window.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::init\_start\_menu()

{

// Start new dialog box.

start\_menu = new class start\_menu();

start\_menu -> setModal(true);

start\_menu->exec();

// Get choice from dialog box.

if(start\_menu->close() == true)

{

menu\_choice = start\_menu->get\_menu\_choice();

}

else

{

menu\_choice = "new";

}

};

/\*\*

\* @brief Starts new wallet using menmonic seed phrase or restoring an old one from a word list.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::init\_wallet()

{

// Create new wallet.

if(menu\_choice == "new")

{

bc::wallet::word\_list mnemonicSeed;

wallet = new Wallet();

}

// Restore existing wallet.

else if(menu\_choice =="restore")

{

get\_mnemonic\_phrase();

wallet = new Wallet(word\_list);

}

};

/\*\*

\* @brief Gets mnemonic phrase from the restore wallet dialog box.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::get\_mnemonic\_phrase()

{

// Starts new dialog window which asks users for 12-word mnemonic phrase.

restore\_wallet = new class restore\_wallet();

// Main focus of UI will be on new restore\_wallet window.

restore\_wallet->setModal(true);

// Once accepted, get the vector containing phrase.

if(restore\_wallet->exec() == QDialog::Accepted){

word\_list = restore\_wallet->get\_word\_list();

}

};

/\*\*

\* @brief Sets the text of the availabe payment address.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::set\_available\_payment\_address()

{

ui->btc\_address->setText(QString::fromStdString(wallet->getAddress(1).encoded()));

};

/\*\*

\* @brief Sets the text of the available balance of the bitcoin wallet.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::set\_btc\_balance()

{

ui->btc\_balance->setText(QString::fromStdString(wallet->get\_balance\_as\_string()));

};

/\*\*

\* @brief Refreshes the main tab including the payment addresses and balance.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::set\_main\_tab()

{

this->set\_available\_payment\_address();

this->set\_btc\_balance();

};

/\*\*

\* @brief Refreshes the analytics tab including the fee recommendations.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::set\_analytics\_tab()

{

// Get network fee recommendations.

network->refreshFeeRecommendations();

// Change text for fee recommendations.

ui->fastwait\_fee->setText(QString::number(network->getFastestFee()) + " Satoshis per Byte");

ui->midwait\_fee->setText(QString::number(network->getHalfHourFee()) + " Satoshis per Byte");

ui->highwait\_fee->setText(QString::number(network->getHourFee()) + " Satoshis per Byte");

};

/\*\*

\* @brief Refreshes the send transaction tab including the maximum fee that user can send.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::set\_send\_tab()

{

// Get network free recommendations.

network->refreshFeeRecommendations();

// Set a maximum fee user can send so that the user is not overpaying a fee.

// This reduces the flexibility of the wallet but places greater important on protecting user from overpaying fees.

ui->fee\_slider->setMaximum(network->getFastestFee());

};

/\*\*

\* @brief Refreshes the history tab by showing all the last transactions involving wallet's addresses.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::set\_history\_tab()

{

// Vector holding last history of transactions.

std::vector< std::tuple<unsigned long long,bc::hash\_digest,int> > tx = wallet-> get\_transaction\_history();

// Number of transactions.

const int NUM\_TX = tx.size();

// Select scrollArea.

QScrollArea \*list\_of\_tx = ui->tx\_scroll\_area;

list\_of\_tx->setBackgroundRole(QPalette::Window);

list\_of\_tx->setFrameShadow(QFrame::Plain);

list\_of\_tx->setFrameShape(QFrame::NoFrame);

list\_of\_tx->setWidgetResizable(true);

// Array of transactions.

QGroupBox \* transactions[NUM\_TX];

// Add transaction widget.

for (int i = 0;i<NUM\_TX;i++)

{

QGroupBox \*groupBox = new QGroupBox(tr("&Transaction"));

QLabel \*date = new QLabel("Block Height: " + QString::number(std::get<2>(tx[i])));

QLabel \*hash = new QLabel("Transaction Hash: " + QString::fromStdString(bc::encode\_hash(std::get<1>(tx[i]))));

//bc::encode\_base10(output.value(), 8)

QLabel \*value = new QLabel("Value: " + QString::fromStdString(bc::encode\_base10(std::get<0>(tx[i]),8)) + " BTC");

QVBoxLayout \*vbox = new QVBoxLayout;

vbox->addWidget(date);

vbox->addWidget(hash);

vbox->addWidget(value);

vbox->addStretch(1);

groupBox->setLayout(vbox);

transactions[i] = groupBox;

}

// Add box to main scroll area widget.

QWidget\* boxArea = new QWidget;

boxArea->setSizePolicy(QSizePolicy::MinimumExpanding, QSizePolicy::MinimumExpanding);

boxArea->setLayout(new QVBoxLayout(boxArea));

list\_of\_tx->setWidget(boxArea);

QLayout \*lay = boxArea->layout();

// Add transactions to boxArea.

for (int i = 0;i<NUM\_TX;i++)

{

lay->addWidget(transactions[i]);

}

};

/\*\*

\* @brief Refreshes the script tab.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::set\_script\_tab()

{

QScrollArea \*script\_console = ui->script\_scrollArea;

script\_console->setBackgroundRole(QPalette::Window);

script\_console->setFrameShadow(QFrame::Plain);

script\_console->setFrameShape(QFrame::NoFrame);

script\_console->setWidgetResizable(true);

QWidget\* boxArea = new QWidget;

boxArea->setSizePolicy(QSizePolicy::MinimumExpanding, QSizePolicy::MinimumExpanding);

boxArea->setLayout(new QVBoxLayout(boxArea));

script\_console->setWidget(boxArea);

script\_layout = boxArea->layout();

};

/\*\*

\* @brief Refreshes a given tab that has been clicked.

\* @param index

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::on\_tabWidget\_tabBarClicked(int index)

{

switch(index)

{

case 0:

//main

this->set\_main\_tab();

break;

case 1:

//send

this->set\_send\_tab();

break;

case 2:

//history

this->set\_history\_tab();

break;

case 3:

// fees

this->set\_analytics\_tab();

break;

case 4:

// script

this->set\_script\_tab();

break;

}

};

/\*\*

\* @brief User clicked on copy bitcoin address. Bitcoin address is copied to computer's clipboard.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::on\_copy\_btc\_address\_clicked()

{

// Access clipboard.

QClipboard \*clipboard = QApplication::clipboard();

clipboard->setText(ui->btc\_address->text());

};

/\*\*

\* @brief User slided fee slider on send transaction tab.

\* @param position

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::on\_fee\_slider\_sliderMoved(int position)

{

ui->sat\_byte\_fee->setText(QString::number(position)+" Satoshis per Byte");

};

/\*\*

\* @brief Performs a check that a given payment address is a legimiate address.

\* @return boolean

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bool

app::is\_valid\_address()

{

std::string address = ui->send\_btc\_address->toPlainText().toStdString();

std::cout << address << std::endl;

/\*\* @todo do appropriate address validation using checksum

\*/

// Bitcoin address is between 26 and 35 characters.

// check length

if(address.length()<26 || address.length() >35)

{

return false;

}

/\*\*

@todo - check base58 encoding

\*/

else

{

return true;

}

};

/\*\*

\* @brief Sends transaction on send button in send transaction tab. Does validation first.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

app::on\_send\_tx\_clicked()

{

// validate transaction

if(this->is\_validate\_tx())

{

//send transaction

this->send\_transaction();

}

};

/\*\*

\* @brief app::is\_validate\_tx

\* @return

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bool

app::is\_validate\_tx()

{

//validate btc address

if(!is\_valid\_address())

{

// Show error message.

error\_msg.showMessage("Invalid address!");

return false;

};

// validate btc amount

if(wallet->getBalance() < (ui->send\_btc\_amount->toPlainText().toInt()\*100000 + ui->fee\_slider->value() ))

{

std::cout << ui->fee\_slider->value() << std::endl;

// Show error message.

error\_msg.showMessage("Not enough money!");

return false;

};

return true;

};

/\*\*

\* @brief Broadcasts transaction to the network.

\* @return

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bool

app::send\_transaction()

{

// Get address, value, and fee.

std::string address = ui->send\_btc\_address->toPlainText().toStdString();

unsigned long long amount = ui->send\_btc\_amount->toPlainText().toInt()\*100000;

unsigned long long fee = ui->fee\_slider->value();

// Broadcast transaction.

wallet->build\_P2PKH(address,amount,fee);

};

/\*\*

\* @brief app::on\_run\_script\_btn\_clicked

\*

\* @author Philip Glazman

\* @date 4/30/18

\*/

void

app::on\_run\_script\_btn\_clicked()

{

this->run\_script();

}

/\*\*

\* @brief app::run\_script

\*

\* @author Philip Glazman

\* @date 4/30/18

\*/

void

app::run\_script()

{

std::string witness = ui->witness\_text\_edit->toPlainText().toStdString();

std::string witness\_script = ui->witness\_script\_text\_edit->toPlainText().toStdString();

script->clear\_script();

script->build\_script(witness,witness\_script);

if(script->is\_valid())

{

this->write\_to\_script\_console("Valid!");

}

else

{

this->write\_to\_script\_console("Error - Script is invalid.");

}

};

/\*\*

\* @brief app::write\_to\_script\_console

\* @param msg

\*

\* @author Philip Glazman

\* @date 5/3/18

\*/

void

app::write\_to\_script\_console(std::string msg)

{

QLabel \*console\_item = new QLabel(QString::fromStdString(msg));

script\_layout->addWidget(console\_item);

};

~~~~~~~~~~~~~~~~~~~~~Source code for file ui\_restore\_wallet.h~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'restore\_wallet.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.10.1

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef UI\_RESTORE\_WALLET\_H

#define UI\_RESTORE\_WALLET\_H

#include <QtCore/QVariant>

#include <QtWidgets/QAction>

#include <QtWidgets/QApplication>

#include <QtWidgets/QButtonGroup>

#include <QtWidgets/QDialog>

#include <QtWidgets/QDialogButtonBox>

#include <QtWidgets/QHBoxLayout>

#include <QtWidgets/QHeaderView>

#include <QtWidgets/QLabel>

#include <QtWidgets/QLineEdit>

#include <QtWidgets/QVBoxLayout>

#include <QtWidgets/QWidget>

QT\_BEGIN\_NAMESPACE

class Ui\_restore\_wallet

{

public:

QDialogButtonBox \*buttonBox;

QLabel \*label\_13;

QWidget \*layoutWidget;

QHBoxLayout \*horizontalLayout\_13;

QVBoxLayout \*verticalLayout;

QHBoxLayout \*horizontalLayout;

QLabel \*label;

QLineEdit \*word\_1;

QHBoxLayout \*horizontalLayout\_2;

QLabel \*label\_2;

QLineEdit \*word\_2;

QHBoxLayout \*horizontalLayout\_3;

QLabel \*label\_3;

QLineEdit \*word\_3;

QHBoxLayout \*horizontalLayout\_4;

QLabel \*label\_4;

QLineEdit \*word\_4;

QHBoxLayout \*horizontalLayout\_5;

QLabel \*label\_5;

QLineEdit \*word\_5;

QHBoxLayout \*horizontalLayout\_6;

QLabel \*label\_6;

QLineEdit \*word\_6;

QVBoxLayout \*verticalLayout\_2;

QHBoxLayout \*horizontalLayout\_7;

QLabel \*label\_7;

QLineEdit \*word\_7;

QHBoxLayout \*horizontalLayout\_8;

QLabel \*label\_8;

QLineEdit \*word\_8;

QHBoxLayout \*horizontalLayout\_9;

QLabel \*label\_9;

QLineEdit \*word\_9;

QHBoxLayout \*horizontalLayout\_10;

QLabel \*label\_10;

QLineEdit \*word\_10;

QHBoxLayout \*horizontalLayout\_11;

QLabel \*label\_11;

QLineEdit \*word\_11;

QHBoxLayout \*horizontalLayout\_12;

QLabel \*label\_12;

QLineEdit \*word\_12;

void setupUi(QDialog \*restore\_wallet)

{

if (restore\_wallet->objectName().isEmpty())

restore\_wallet->setObjectName(QStringLiteral("restore\_wallet"));

restore\_wallet->resize(400, 300);

buttonBox = new QDialogButtonBox(restore\_wallet);

buttonBox->setObjectName(QStringLiteral("buttonBox"));

buttonBox->setGeometry(QRect(30, 250, 341, 32));

buttonBox->setOrientation(Qt::Horizontal);

buttonBox->setStandardButtons(QDialogButtonBox::Cancel|QDialogButtonBox::Ok);

buttonBox->setCenterButtons(false);

label\_13 = new QLabel(restore\_wallet);

label\_13->setObjectName(QStringLiteral("label\_13"));

label\_13->setGeometry(QRect(130, 10, 141, 16));

layoutWidget = new QWidget(restore\_wallet);

layoutWidget->setObjectName(QStringLiteral("layoutWidget"));

layoutWidget->setGeometry(QRect(40, 30, 312, 218));

layoutWidget->setAutoFillBackground(false);

layoutWidget->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_13 = new QHBoxLayout(layoutWidget);

horizontalLayout\_13->setObjectName(QStringLiteral("horizontalLayout\_13"));

horizontalLayout\_13->setContentsMargins(0, 0, 0, 0);

verticalLayout = new QVBoxLayout();

verticalLayout->setObjectName(QStringLiteral("verticalLayout"));

horizontalLayout = new QHBoxLayout();

horizontalLayout->setObjectName(QStringLiteral("horizontalLayout"));

label = new QLabel(layoutWidget);

label->setObjectName(QStringLiteral("label"));

label->setAutoFillBackground(false);

label->setInputMethodHints(Qt::ImhNone);

horizontalLayout->addWidget(label);

word\_1 = new QLineEdit(layoutWidget);

word\_1->setObjectName(QStringLiteral("word\_1"));

word\_1->setAutoFillBackground(false);

word\_1->setInputMethodHints(Qt::ImhNone);

word\_1->setMaxLength(50);

word\_1->setClearButtonEnabled(false);

horizontalLayout->addWidget(word\_1);

verticalLayout->addLayout(horizontalLayout);

horizontalLayout\_2 = new QHBoxLayout();

horizontalLayout\_2->setObjectName(QStringLiteral("horizontalLayout\_2"));

label\_2 = new QLabel(layoutWidget);

label\_2->setObjectName(QStringLiteral("label\_2"));

label\_2->setAutoFillBackground(false);

label\_2->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_2->addWidget(label\_2);

word\_2 = new QLineEdit(layoutWidget);

word\_2->setObjectName(QStringLiteral("word\_2"));

word\_2->setAutoFillBackground(false);

word\_2->setInputMethodHints(Qt::ImhNone);

word\_2->setMaxLength(50);

word\_2->setClearButtonEnabled(false);

horizontalLayout\_2->addWidget(word\_2);

verticalLayout->addLayout(horizontalLayout\_2);

horizontalLayout\_3 = new QHBoxLayout();

horizontalLayout\_3->setObjectName(QStringLiteral("horizontalLayout\_3"));

label\_3 = new QLabel(layoutWidget);

label\_3->setObjectName(QStringLiteral("label\_3"));

label\_3->setAutoFillBackground(false);

label\_3->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_3->addWidget(label\_3);

word\_3 = new QLineEdit(layoutWidget);

word\_3->setObjectName(QStringLiteral("word\_3"));

word\_3->setAutoFillBackground(false);

word\_3->setInputMethodHints(Qt::ImhNone);

word\_3->setMaxLength(50);

word\_3->setClearButtonEnabled(false);

horizontalLayout\_3->addWidget(word\_3);

verticalLayout->addLayout(horizontalLayout\_3);

horizontalLayout\_4 = new QHBoxLayout();

horizontalLayout\_4->setObjectName(QStringLiteral("horizontalLayout\_4"));

label\_4 = new QLabel(layoutWidget);

label\_4->setObjectName(QStringLiteral("label\_4"));

label\_4->setAutoFillBackground(false);

label\_4->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_4->addWidget(label\_4);

word\_4 = new QLineEdit(layoutWidget);

word\_4->setObjectName(QStringLiteral("word\_4"));

word\_4->setAutoFillBackground(false);

word\_4->setInputMethodHints(Qt::ImhNone);

word\_4->setMaxLength(50);

word\_4->setClearButtonEnabled(false);

horizontalLayout\_4->addWidget(word\_4);

verticalLayout->addLayout(horizontalLayout\_4);

horizontalLayout\_5 = new QHBoxLayout();

horizontalLayout\_5->setObjectName(QStringLiteral("horizontalLayout\_5"));

label\_5 = new QLabel(layoutWidget);

label\_5->setObjectName(QStringLiteral("label\_5"));

label\_5->setAutoFillBackground(false);

label\_5->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_5->addWidget(label\_5);

word\_5 = new QLineEdit(layoutWidget);

word\_5->setObjectName(QStringLiteral("word\_5"));

word\_5->setAutoFillBackground(false);

word\_5->setInputMethodHints(Qt::ImhNone);

word\_5->setMaxLength(50);

word\_5->setClearButtonEnabled(false);

horizontalLayout\_5->addWidget(word\_5);

verticalLayout->addLayout(horizontalLayout\_5);

horizontalLayout\_6 = new QHBoxLayout();

horizontalLayout\_6->setObjectName(QStringLiteral("horizontalLayout\_6"));

label\_6 = new QLabel(layoutWidget);

label\_6->setObjectName(QStringLiteral("label\_6"));

label\_6->setAutoFillBackground(false);

label\_6->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_6->addWidget(label\_6);

word\_6 = new QLineEdit(layoutWidget);

word\_6->setObjectName(QStringLiteral("word\_6"));

word\_6->setAutoFillBackground(false);

word\_6->setInputMethodHints(Qt::ImhNone);

word\_6->setMaxLength(50);

word\_6->setClearButtonEnabled(false);

horizontalLayout\_6->addWidget(word\_6);

verticalLayout->addLayout(horizontalLayout\_6);

horizontalLayout\_13->addLayout(verticalLayout);

verticalLayout\_2 = new QVBoxLayout();

verticalLayout\_2->setObjectName(QStringLiteral("verticalLayout\_2"));

horizontalLayout\_7 = new QHBoxLayout();

horizontalLayout\_7->setObjectName(QStringLiteral("horizontalLayout\_7"));

label\_7 = new QLabel(layoutWidget);

label\_7->setObjectName(QStringLiteral("label\_7"));

label\_7->setAutoFillBackground(false);

label\_7->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_7->addWidget(label\_7);

word\_7 = new QLineEdit(layoutWidget);

word\_7->setObjectName(QStringLiteral("word\_7"));

word\_7->setAutoFillBackground(false);

word\_7->setInputMethodHints(Qt::ImhNone);

word\_7->setMaxLength(50);

word\_7->setClearButtonEnabled(false);

horizontalLayout\_7->addWidget(word\_7);

verticalLayout\_2->addLayout(horizontalLayout\_7);

horizontalLayout\_8 = new QHBoxLayout();

horizontalLayout\_8->setObjectName(QStringLiteral("horizontalLayout\_8"));

label\_8 = new QLabel(layoutWidget);

label\_8->setObjectName(QStringLiteral("label\_8"));

label\_8->setAutoFillBackground(false);

label\_8->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_8->addWidget(label\_8);

word\_8 = new QLineEdit(layoutWidget);

word\_8->setObjectName(QStringLiteral("word\_8"));

word\_8->setAutoFillBackground(false);

word\_8->setInputMethodHints(Qt::ImhNone);

word\_8->setMaxLength(50);

word\_8->setClearButtonEnabled(false);

horizontalLayout\_8->addWidget(word\_8);

verticalLayout\_2->addLayout(horizontalLayout\_8);

horizontalLayout\_9 = new QHBoxLayout();

horizontalLayout\_9->setObjectName(QStringLiteral("horizontalLayout\_9"));

label\_9 = new QLabel(layoutWidget);

label\_9->setObjectName(QStringLiteral("label\_9"));

label\_9->setAutoFillBackground(false);

label\_9->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_9->addWidget(label\_9);

word\_9 = new QLineEdit(layoutWidget);

word\_9->setObjectName(QStringLiteral("word\_9"));

word\_9->setAutoFillBackground(false);

word\_9->setInputMethodHints(Qt::ImhNone);

word\_9->setMaxLength(50);

word\_9->setClearButtonEnabled(false);

horizontalLayout\_9->addWidget(word\_9);

verticalLayout\_2->addLayout(horizontalLayout\_9);

horizontalLayout\_10 = new QHBoxLayout();

horizontalLayout\_10->setObjectName(QStringLiteral("horizontalLayout\_10"));

label\_10 = new QLabel(layoutWidget);

label\_10->setObjectName(QStringLiteral("label\_10"));

label\_10->setAutoFillBackground(false);

label\_10->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_10->addWidget(label\_10);

word\_10 = new QLineEdit(layoutWidget);

word\_10->setObjectName(QStringLiteral("word\_10"));

word\_10->setAutoFillBackground(false);

word\_10->setInputMethodHints(Qt::ImhNone);

word\_10->setMaxLength(50);

word\_10->setClearButtonEnabled(false);

horizontalLayout\_10->addWidget(word\_10);

verticalLayout\_2->addLayout(horizontalLayout\_10);

horizontalLayout\_11 = new QHBoxLayout();

horizontalLayout\_11->setObjectName(QStringLiteral("horizontalLayout\_11"));

label\_11 = new QLabel(layoutWidget);

label\_11->setObjectName(QStringLiteral("label\_11"));

label\_11->setAutoFillBackground(false);

label\_11->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_11->addWidget(label\_11);

word\_11 = new QLineEdit(layoutWidget);

word\_11->setObjectName(QStringLiteral("word\_11"));

word\_11->setAutoFillBackground(false);

word\_11->setInputMethodHints(Qt::ImhNone);

word\_11->setMaxLength(50);

word\_11->setClearButtonEnabled(false);

horizontalLayout\_11->addWidget(word\_11);

verticalLayout\_2->addLayout(horizontalLayout\_11);

horizontalLayout\_12 = new QHBoxLayout();

horizontalLayout\_12->setObjectName(QStringLiteral("horizontalLayout\_12"));

label\_12 = new QLabel(layoutWidget);

label\_12->setObjectName(QStringLiteral("label\_12"));

label\_12->setAutoFillBackground(false);

label\_12->setInputMethodHints(Qt::ImhNone);

horizontalLayout\_12->addWidget(label\_12);

word\_12 = new QLineEdit(layoutWidget);

word\_12->setObjectName(QStringLiteral("word\_12"));

word\_12->setAutoFillBackground(false);

word\_12->setInputMethodHints(Qt::ImhNone);

word\_12->setMaxLength(50);

word\_12->setClearButtonEnabled(false);

horizontalLayout\_12->addWidget(word\_12);

verticalLayout\_2->addLayout(horizontalLayout\_12);

horizontalLayout\_13->addLayout(verticalLayout\_2);

retranslateUi(restore\_wallet);

QObject::connect(buttonBox, SIGNAL(accepted()), restore\_wallet, SLOT(accept()));

QObject::connect(buttonBox, SIGNAL(rejected()), restore\_wallet, SLOT(reject()));

QMetaObject::connectSlotsByName(restore\_wallet);

} // setupUi

void retranslateUi(QDialog \*restore\_wallet)

{

restore\_wallet->setWindowTitle(QApplication::translate("restore\_wallet", "Atlas", nullptr));

label\_13->setText(QApplication::translate("restore\_wallet", "Enter 12-word phrase:", nullptr));

label->setText(QApplication::translate("restore\_wallet", "1.", nullptr));

word\_1->setPlaceholderText(QString());

label\_2->setText(QApplication::translate("restore\_wallet", "2.", nullptr));

word\_2->setPlaceholderText(QString());

label\_3->setText(QApplication::translate("restore\_wallet", "3.", nullptr));

word\_3->setPlaceholderText(QString());

label\_4->setText(QApplication::translate("restore\_wallet", "4.", nullptr));

word\_4->setPlaceholderText(QString());

label\_5->setText(QApplication::translate("restore\_wallet", "5.", nullptr));

word\_5->setPlaceholderText(QString());

label\_6->setText(QApplication::translate("restore\_wallet", "6.", nullptr));

word\_6->setPlaceholderText(QString());

label\_7->setText(QApplication::translate("restore\_wallet", "7.", nullptr));

word\_7->setPlaceholderText(QString());

label\_8->setText(QApplication::translate("restore\_wallet", "8.", nullptr));

word\_8->setPlaceholderText(QString());

label\_9->setText(QApplication::translate("restore\_wallet", "9.", nullptr));

word\_9->setPlaceholderText(QString());

label\_10->setText(QApplication::translate("restore\_wallet", "10.", nullptr));

word\_10->setPlaceholderText(QString());

label\_11->setText(QApplication::translate("restore\_wallet", "11.", nullptr));

word\_11->setPlaceholderText(QString());

label\_12->setText(QApplication::translate("restore\_wallet", "12.", nullptr));

word\_12->setPlaceholderText(QString());

} // retranslateUi

};

namespace Ui {

class restore\_wallet: public Ui\_restore\_wallet {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // UI\_RESTORE\_WALLET\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file moc\_app.cpp~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Meta object code from reading C++ file 'app.h'

\*\*

\*\* Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)

\*\*

\*\* WARNING! All changes made in this file will be lost!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "../Atlas/app.h"

#include <QtCore/qbytearray.h>

#include <QtCore/qmetatype.h>

#if !defined(Q\_MOC\_OUTPUT\_REVISION)

#error "The header file 'app.h' doesn't include <QObject>."

#elif Q\_MOC\_OUTPUT\_REVISION != 67

#error "This file was generated using the moc from 5.10.1. It"

#error "cannot be used with the include files from this version of Qt."

#error "(The moc has changed too much.)"

#endif

QT\_BEGIN\_MOC\_NAMESPACE

QT\_WARNING\_PUSH

QT\_WARNING\_DISABLE\_DEPRECATED

struct qt\_meta\_stringdata\_app\_t {

QByteArrayData data[9];

char stringdata0[146];

};

#define QT\_MOC\_LITERAL(idx, ofs, len) \

Q\_STATIC\_BYTE\_ARRAY\_DATA\_HEADER\_INITIALIZER\_WITH\_OFFSET(len, \

qptrdiff(offsetof(qt\_meta\_stringdata\_app\_t, stringdata0) + ofs \

- idx \* sizeof(QByteArrayData)) \

)

static const qt\_meta\_stringdata\_app\_t qt\_meta\_stringdata\_app = {

{

QT\_MOC\_LITERAL(0, 0, 3), // "app"

QT\_MOC\_LITERAL(1, 4, 26), // "on\_tabWidget\_tabBarClicked"

QT\_MOC\_LITERAL(2, 31, 0), // ""

QT\_MOC\_LITERAL(3, 32, 5), // "index"

QT\_MOC\_LITERAL(4, 38, 27), // "on\_copy\_btc\_address\_clicked"

QT\_MOC\_LITERAL(5, 66, 25), // "on\_fee\_slider\_sliderMoved"

QT\_MOC\_LITERAL(6, 92, 8), // "position"

QT\_MOC\_LITERAL(7, 101, 18), // "on\_send\_tx\_clicked"

QT\_MOC\_LITERAL(8, 120, 25) // "on\_run\_script\_btn\_clicked"

},

"app\0on\_tabWidget\_tabBarClicked\0\0index\0"

"on\_copy\_btc\_address\_clicked\0"

"on\_fee\_slider\_sliderMoved\0position\0"

"on\_send\_tx\_clicked\0on\_run\_script\_btn\_clicked"

};

#undef QT\_MOC\_LITERAL

static const uint qt\_meta\_data\_app[] = {

// content:

7, // revision

0, // classname

0, 0, // classinfo

5, 14, // methods

0, 0, // properties

0, 0, // enums/sets

0, 0, // constructors

0, // flags

0, // signalCount

// slots: name, argc, parameters, tag, flags

1, 1, 39, 2, 0x08 /\* Private \*/,

4, 0, 42, 2, 0x08 /\* Private \*/,

5, 1, 43, 2, 0x08 /\* Private \*/,

7, 0, 46, 2, 0x08 /\* Private \*/,

8, 0, 47, 2, 0x08 /\* Private \*/,

// slots: parameters

QMetaType::Void, QMetaType::Int, 3,

QMetaType::Void,

QMetaType::Void, QMetaType::Int, 6,

QMetaType::Void,

QMetaType::Void,

0 // eod

};

void app::qt\_static\_metacall(QObject \*\_o, QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

if (\_c == QMetaObject::InvokeMetaMethod) {

app \*\_t = static\_cast<app \*>(\_o);

Q\_UNUSED(\_t)

switch (\_id) {

case 0: \_t->on\_tabWidget\_tabBarClicked((\*reinterpret\_cast< int(\*)>(\_a[1]))); break;

case 1: \_t->on\_copy\_btc\_address\_clicked(); break;

case 2: \_t->on\_fee\_slider\_sliderMoved((\*reinterpret\_cast< int(\*)>(\_a[1]))); break;

case 3: \_t->on\_send\_tx\_clicked(); break;

case 4: \_t->on\_run\_script\_btn\_clicked(); break;

default: ;

}

}

}

QT\_INIT\_METAOBJECT const QMetaObject app::staticMetaObject = {

{ &QMainWindow::staticMetaObject, qt\_meta\_stringdata\_app.data,

qt\_meta\_data\_app, qt\_static\_metacall, nullptr, nullptr}

};

const QMetaObject \*app::metaObject() const

{

return QObject::d\_ptr->metaObject ? QObject::d\_ptr->dynamicMetaObject() : &staticMetaObject;

}

void \*app::qt\_metacast(const char \*\_clname)

{

if (!\_clname) return nullptr;

if (!strcmp(\_clname, qt\_meta\_stringdata\_app.stringdata0))

return static\_cast<void\*>(this);

return QMainWindow::qt\_metacast(\_clname);

}

int app::qt\_metacall(QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

\_id = QMainWindow::qt\_metacall(\_c, \_id, \_a);

if (\_id < 0)

return \_id;

if (\_c == QMetaObject::InvokeMetaMethod) {

if (\_id < 5)

qt\_static\_metacall(this, \_c, \_id, \_a);

\_id -= 5;

} else if (\_c == QMetaObject::RegisterMethodArgumentMetaType) {

if (\_id < 5)

\*reinterpret\_cast<int\*>(\_a[0]) = -1;

\_id -= 5;

}

return \_id;

}

QT\_WARNING\_POP

QT\_END\_MOC\_NAMESPACE

~~~~~~~~~~~~~~~~~~~~~Source code for file ui\_start\_menu.h~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'start\_menu.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.10.1

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef UI\_START\_MENU\_H

#define UI\_START\_MENU\_H

#include <QtCore/QVariant>

#include <QtWidgets/QAction>

#include <QtWidgets/QApplication>

#include <QtWidgets/QButtonGroup>

#include <QtWidgets/QDialog>

#include <QtWidgets/QHeaderView>

#include <QtWidgets/QPushButton>

#include <QtWidgets/QVBoxLayout>

#include <QtWidgets/QWidget>

QT\_BEGIN\_NAMESPACE

class Ui\_start\_menu

{

public:

QWidget \*layoutWidget;

QVBoxLayout \*verticalLayout;

QPushButton \*create\_new\_wallet;

QPushButton \*restore\_existing\_wallet;

void setupUi(QDialog \*start\_menu)

{

if (start\_menu->objectName().isEmpty())

start\_menu->setObjectName(QStringLiteral("start\_menu"));

start\_menu->resize(400, 300);

start\_menu->setAutoFillBackground(false);

layoutWidget = new QWidget(start\_menu);

layoutWidget->setObjectName(QStringLiteral("layoutWidget"));

layoutWidget->setGeometry(QRect(40, 80, 291, 111));

verticalLayout = new QVBoxLayout(layoutWidget);

verticalLayout->setObjectName(QStringLiteral("verticalLayout"));

verticalLayout->setContentsMargins(0, 0, 0, 0);

create\_new\_wallet = new QPushButton(layoutWidget);

create\_new\_wallet->setObjectName(QStringLiteral("create\_new\_wallet"));

verticalLayout->addWidget(create\_new\_wallet);

restore\_existing\_wallet = new QPushButton(layoutWidget);

restore\_existing\_wallet->setObjectName(QStringLiteral("restore\_existing\_wallet"));

verticalLayout->addWidget(restore\_existing\_wallet);

retranslateUi(start\_menu);

QMetaObject::connectSlotsByName(start\_menu);

} // setupUi

void retranslateUi(QDialog \*start\_menu)

{

start\_menu->setWindowTitle(QApplication::translate("start\_menu", "Atlas", nullptr));

create\_new\_wallet->setText(QApplication::translate("start\_menu", "Create New Wallet", nullptr));

restore\_existing\_wallet->setText(QApplication::translate("start\_menu", "Restore Existing Wallet", nullptr));

} // retranslateUi

};

namespace Ui {

class start\_menu: public Ui\_start\_menu {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // UI\_START\_MENU\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file ui\_error.h~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'error.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.10.1

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef UI\_ERROR\_H

#define UI\_ERROR\_H

#include <QtCore/QVariant>

#include <QtWidgets/QAction>

#include <QtWidgets/QApplication>

#include <QtWidgets/QButtonGroup>

#include <QtWidgets/QDialog>

#include <QtWidgets/QDialogButtonBox>

#include <QtWidgets/QHeaderView>

QT\_BEGIN\_NAMESPACE

class Ui\_Dialog

{

public:

QDialogButtonBox \*buttonBox;

void setupUi(QDialog \*Dialog)

{

if (Dialog->objectName().isEmpty())

Dialog->setObjectName(QStringLiteral("Dialog"));

Dialog->resize(400, 300);

buttonBox = new QDialogButtonBox(Dialog);

buttonBox->setObjectName(QStringLiteral("buttonBox"));

buttonBox->setGeometry(QRect(30, 240, 341, 32));

buttonBox->setOrientation(Qt::Horizontal);

buttonBox->setStandardButtons(QDialogButtonBox::Cancel|QDialogButtonBox::Ok);

retranslateUi(Dialog);

QObject::connect(buttonBox, SIGNAL(accepted()), Dialog, SLOT(accept()));

QObject::connect(buttonBox, SIGNAL(rejected()), Dialog, SLOT(reject()));

QMetaObject::connectSlotsByName(Dialog);

} // setupUi

void retranslateUi(QDialog \*Dialog)

{

Dialog->setWindowTitle(QApplication::translate("Dialog", "Dialog", nullptr));

} // retranslateUi

};

namespace Ui {

class Dialog: public Ui\_Dialog {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // UI\_ERROR\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file moc\_start\_menu.cpp~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Meta object code from reading C++ file 'start\_menu.h'

\*\*

\*\* Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)

\*\*

\*\* WARNING! All changes made in this file will be lost!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "../Atlas/start\_menu.h"

#include <QtCore/qbytearray.h>

#include <QtCore/qmetatype.h>

#if !defined(Q\_MOC\_OUTPUT\_REVISION)

#error "The header file 'start\_menu.h' doesn't include <QObject>."

#elif Q\_MOC\_OUTPUT\_REVISION != 67

#error "This file was generated using the moc from 5.10.1. It"

#error "cannot be used with the include files from this version of Qt."

#error "(The moc has changed too much.)"

#endif

QT\_BEGIN\_MOC\_NAMESPACE

QT\_WARNING\_PUSH

QT\_WARNING\_DISABLE\_DEPRECATED

struct qt\_meta\_stringdata\_start\_menu\_t {

QByteArrayData data[4];

char stringdata0[76];

};

#define QT\_MOC\_LITERAL(idx, ofs, len) \

Q\_STATIC\_BYTE\_ARRAY\_DATA\_HEADER\_INITIALIZER\_WITH\_OFFSET(len, \

qptrdiff(offsetof(qt\_meta\_stringdata\_start\_menu\_t, stringdata0) + ofs \

- idx \* sizeof(QByteArrayData)) \

)

static const qt\_meta\_stringdata\_start\_menu\_t qt\_meta\_stringdata\_start\_menu = {

{

QT\_MOC\_LITERAL(0, 0, 10), // "start\_menu"

QT\_MOC\_LITERAL(1, 11, 28), // "on\_create\_new\_wallet\_clicked"

QT\_MOC\_LITERAL(2, 40, 0), // ""

QT\_MOC\_LITERAL(3, 41, 34) // "on\_restore\_existing\_wallet\_cl..."

},

"start\_menu\0on\_create\_new\_wallet\_clicked\0"

"\0on\_restore\_existing\_wallet\_clicked"

};

#undef QT\_MOC\_LITERAL

static const uint qt\_meta\_data\_start\_menu[] = {

// content:

7, // revision

0, // classname

0, 0, // classinfo

2, 14, // methods

0, 0, // properties

0, 0, // enums/sets

0, 0, // constructors

0, // flags

0, // signalCount

// slots: name, argc, parameters, tag, flags

1, 0, 24, 2, 0x08 /\* Private \*/,

3, 0, 25, 2, 0x08 /\* Private \*/,

// slots: parameters

QMetaType::Void,

QMetaType::Void,

0 // eod

};

void start\_menu::qt\_static\_metacall(QObject \*\_o, QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

if (\_c == QMetaObject::InvokeMetaMethod) {

start\_menu \*\_t = static\_cast<start\_menu \*>(\_o);

Q\_UNUSED(\_t)

switch (\_id) {

case 0: \_t->on\_create\_new\_wallet\_clicked(); break;

case 1: \_t->on\_restore\_existing\_wallet\_clicked(); break;

default: ;

}

}

Q\_UNUSED(\_a);

}

QT\_INIT\_METAOBJECT const QMetaObject start\_menu::staticMetaObject = {

{ &QDialog::staticMetaObject, qt\_meta\_stringdata\_start\_menu.data,

qt\_meta\_data\_start\_menu, qt\_static\_metacall, nullptr, nullptr}

};

const QMetaObject \*start\_menu::metaObject() const

{

return QObject::d\_ptr->metaObject ? QObject::d\_ptr->dynamicMetaObject() : &staticMetaObject;

}

void \*start\_menu::qt\_metacast(const char \*\_clname)

{

if (!\_clname) return nullptr;

if (!strcmp(\_clname, qt\_meta\_stringdata\_start\_menu.stringdata0))

return static\_cast<void\*>(this);

return QDialog::qt\_metacast(\_clname);

}

int start\_menu::qt\_metacall(QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

\_id = QDialog::qt\_metacall(\_c, \_id, \_a);

if (\_id < 0)

return \_id;

if (\_c == QMetaObject::InvokeMetaMethod) {

if (\_id < 2)

qt\_static\_metacall(this, \_c, \_id, \_a);

\_id -= 2;

} else if (\_c == QMetaObject::RegisterMethodArgumentMetaType) {

if (\_id < 2)

\*reinterpret\_cast<int\*>(\_a[0]) = -1;

\_id -= 2;

}

return \_id;

}

QT\_WARNING\_POP

QT\_END\_MOC\_NAMESPACE

~~~~~~~~~~~~~~~~~~~~~Source code for file ui\_app.h~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Form generated from reading UI file 'app.ui'

\*\*

\*\* Created by: Qt User Interface Compiler version 5.10.1

\*\*

\*\* WARNING! All changes made in this file will be lost when recompiling UI file!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#ifndef UI\_APP\_H

#define UI\_APP\_H

#include <QtCore/QVariant>

#include <QtWidgets/QAction>

#include <QtWidgets/QApplication>

#include <QtWidgets/QButtonGroup>

#include <QtWidgets/QGroupBox>

#include <QtWidgets/QHeaderView>

#include <QtWidgets/QLabel>

#include <QtWidgets/QMainWindow>

#include <QtWidgets/QMenu>

#include <QtWidgets/QMenuBar>

#include <QtWidgets/QPushButton>

#include <QtWidgets/QScrollArea>

#include <QtWidgets/QScrollBar>

#include <QtWidgets/QTabWidget>

#include <QtWidgets/QTextEdit>

#include <QtWidgets/QWidget>

QT\_BEGIN\_NAMESPACE

class Ui\_app

{

public:

QAction \*actionExit;

QWidget \*centralWidget;

QTabWidget \*tabWidget;

QWidget \*tab\_main;

QGroupBox \*groupBox;

QLabel \*label;

QLabel \*label\_2;

QLabel \*label\_3;

QLabel \*btc\_recieved;

QLabel \*btc\_sent;

QLabel \*btc\_balance;

QGroupBox \*groupBox\_2;

QLabel \*btc\_address;

QPushButton \*copy\_btc\_address;

QWidget \*tab\_send;

QLabel \*label\_4;

QLabel \*label\_5;

QLabel \*label\_6;

QPushButton \*send\_tx;

QPushButton \*pushButton\_2;

QTextEdit \*send\_btc\_address;

QTextEdit \*send\_btc\_amount;

QScrollBar \*fee\_slider;

QLabel \*sat\_byte\_fee;

QLabel \*label\_7;

QWidget \*tab\_history;

QScrollArea \*tx\_scroll\_area;

QWidget \*scrollAreaWidgetContents;

QWidget \*tab\_analytics;

QGroupBox \*groupBox\_3;

QLabel \*label\_8;

QLabel \*label\_9;

QLabel \*label\_10;

QLabel \*fastwait\_fee;

QLabel \*midwait\_fee;

QLabel \*highwait\_fee;

QWidget \*tab\_script;

QGroupBox \*witness\_box;

QTextEdit \*witness\_text\_edit;

QGroupBox \*witness\_script\_box;

QTextEdit \*witness\_script\_text\_edit;

QPushButton \*run\_script\_btn;

QScrollArea \*script\_scrollArea;

QWidget \*scrollAreaWidgetContents\_2;

QLabel \*label\_11;

QMenuBar \*menuBar;

QMenu \*menuAtlas;

QMenu \*menuHistory;

QMenu \*menuAnalytics;

QMenu \*menuScript;

void setupUi(QMainWindow \*app)

{

if (app->objectName().isEmpty())

app->setObjectName(QStringLiteral("app"));

app->resize(953, 467);

app->setTabShape(QTabWidget::Rounded);

actionExit = new QAction(app);

actionExit->setObjectName(QStringLiteral("actionExit"));

centralWidget = new QWidget(app);

centralWidget->setObjectName(QStringLiteral("centralWidget"));

tabWidget = new QTabWidget(centralWidget);

tabWidget->setObjectName(QStringLiteral("tabWidget"));

tabWidget->setGeometry(QRect(10, 30, 931, 401));

tabWidget->setTabsClosable(false);

tab\_main = new QWidget();

tab\_main->setObjectName(QStringLiteral("tab\_main"));

groupBox = new QGroupBox(tab\_main);

groupBox->setObjectName(QStringLiteral("groupBox"));

groupBox->setGeometry(QRect(260, 20, 401, 191));

groupBox->setAutoFillBackground(false);

label = new QLabel(groupBox);

label->setObjectName(QStringLiteral("label"));

label->setGeometry(QRect(20, 40, 171, 16));

label\_2 = new QLabel(groupBox);

label\_2->setObjectName(QStringLiteral("label\_2"));

label\_2->setGeometry(QRect(20, 70, 171, 16));

label\_3 = new QLabel(groupBox);

label\_3->setObjectName(QStringLiteral("label\_3"));

label\_3->setGeometry(QRect(20, 100, 171, 16));

btc\_recieved = new QLabel(groupBox);

btc\_recieved->setObjectName(QStringLiteral("btc\_recieved"));

btc\_recieved->setGeometry(QRect(210, 40, 171, 20));

btc\_sent = new QLabel(groupBox);

btc\_sent->setObjectName(QStringLiteral("btc\_sent"));

btc\_sent->setGeometry(QRect(210, 70, 181, 16));

btc\_balance = new QLabel(groupBox);

btc\_balance->setObjectName(QStringLiteral("btc\_balance"));

btc\_balance->setGeometry(QRect(210, 100, 181, 16));

groupBox\_2 = new QGroupBox(tab\_main);

groupBox\_2->setObjectName(QStringLiteral("groupBox\_2"));

groupBox\_2->setGeometry(QRect(260, 240, 391, 80));

btc\_address = new QLabel(groupBox\_2);

btc\_address->setObjectName(QStringLiteral("btc\_address"));

btc\_address->setGeometry(QRect(10, 30, 341, 16));

copy\_btc\_address = new QPushButton(groupBox\_2);

copy\_btc\_address->setObjectName(QStringLiteral("copy\_btc\_address"));

copy\_btc\_address->setGeometry(QRect(270, 50, 113, 32));

tabWidget->addTab(tab\_main, QString());

tab\_send = new QWidget();

tab\_send->setObjectName(QStringLiteral("tab\_send"));

label\_4 = new QLabel(tab\_send);

label\_4->setObjectName(QStringLiteral("label\_4"));

label\_4->setGeometry(QRect(30, 40, 81, 16));

label\_5 = new QLabel(tab\_send);

label\_5->setObjectName(QStringLiteral("label\_5"));

label\_5->setGeometry(QRect(30, 80, 81, 16));

label\_6 = new QLabel(tab\_send);

label\_6->setObjectName(QStringLiteral("label\_6"));

label\_6->setGeometry(QRect(30, 120, 81, 16));

send\_tx = new QPushButton(tab\_send);

send\_tx->setObjectName(QStringLiteral("send\_tx"));

send\_tx->setGeometry(QRect(140, 150, 113, 32));

pushButton\_2 = new QPushButton(tab\_send);

pushButton\_2->setObjectName(QStringLiteral("pushButton\_2"));

pushButton\_2->setGeometry(QRect(20, 150, 113, 32));

send\_btc\_address = new QTextEdit(tab\_send);

send\_btc\_address->setObjectName(QStringLiteral("send\_btc\_address"));

send\_btc\_address->setGeometry(QRect(120, 40, 481, 21));

send\_btc\_address->setInputMethodHints(Qt::ImhNone);

send\_btc\_amount = new QTextEdit(tab\_send);

send\_btc\_amount->setObjectName(QStringLiteral("send\_btc\_amount"));

send\_btc\_amount->setGeometry(QRect(120, 80, 101, 21));

fee\_slider = new QScrollBar(tab\_send);

fee\_slider->setObjectName(QStringLiteral("fee\_slider"));

fee\_slider->setGeometry(QRect(120, 120, 160, 16));

fee\_slider->setMaximum(99);

fee\_slider->setOrientation(Qt::Horizontal);

sat\_byte\_fee = new QLabel(tab\_send);

sat\_byte\_fee->setObjectName(QStringLiteral("sat\_byte\_fee"));

sat\_byte\_fee->setGeometry(QRect(300, 120, 131, 16));

label\_7 = new QLabel(tab\_send);

label\_7->setObjectName(QStringLiteral("label\_7"));

label\_7->setGeometry(QRect(230, 80, 81, 16));

tabWidget->addTab(tab\_send, QString());

tab\_history = new QWidget();

tab\_history->setObjectName(QStringLiteral("tab\_history"));

tx\_scroll\_area = new QScrollArea(tab\_history);

tx\_scroll\_area->setObjectName(QStringLiteral("tx\_scroll\_area"));

tx\_scroll\_area->setGeometry(QRect(20, 20, 891, 341));

QSizePolicy sizePolicy(QSizePolicy::Fixed, QSizePolicy::Expanding);

sizePolicy.setHorizontalStretch(0);

sizePolicy.setVerticalStretch(100);

sizePolicy.setHeightForWidth(tx\_scroll\_area->sizePolicy().hasHeightForWidth());

tx\_scroll\_area->setSizePolicy(sizePolicy);

tx\_scroll\_area->setLineWidth(1);

tx\_scroll\_area->setVerticalScrollBarPolicy(Qt::ScrollBarAlwaysOn);

tx\_scroll\_area->setHorizontalScrollBarPolicy(Qt::ScrollBarAlwaysOff);

tx\_scroll\_area->setSizeAdjustPolicy(QAbstractScrollArea::AdjustIgnored);

tx\_scroll\_area->setWidgetResizable(true);

scrollAreaWidgetContents = new QWidget();

scrollAreaWidgetContents->setObjectName(QStringLiteral("scrollAreaWidgetContents"));

scrollAreaWidgetContents->setGeometry(QRect(0, 0, 873, 339));

tx\_scroll\_area->setWidget(scrollAreaWidgetContents);

tabWidget->addTab(tab\_history, QString());

tab\_analytics = new QWidget();

tab\_analytics->setObjectName(QStringLiteral("tab\_analytics"));

groupBox\_3 = new QGroupBox(tab\_analytics);

groupBox\_3->setObjectName(QStringLiteral("groupBox\_3"));

groupBox\_3->setGeometry(QRect(480, 60, 361, 241));

label\_8 = new QLabel(groupBox\_3);

label\_8->setObjectName(QStringLiteral("label\_8"));

label\_8->setGeometry(QRect(20, 40, 161, 16));

label\_9 = new QLabel(groupBox\_3);

label\_9->setObjectName(QStringLiteral("label\_9"));

label\_9->setGeometry(QRect(20, 80, 161, 16));

label\_10 = new QLabel(groupBox\_3);

label\_10->setObjectName(QStringLiteral("label\_10"));

label\_10->setGeometry(QRect(20, 120, 161, 16));

fastwait\_fee = new QLabel(groupBox\_3);

fastwait\_fee->setObjectName(QStringLiteral("fastwait\_fee"));

fastwait\_fee->setGeometry(QRect(180, 40, 161, 16));

midwait\_fee = new QLabel(groupBox\_3);

midwait\_fee->setObjectName(QStringLiteral("midwait\_fee"));

midwait\_fee->setGeometry(QRect(180, 80, 161, 16));

highwait\_fee = new QLabel(groupBox\_3);

highwait\_fee->setObjectName(QStringLiteral("highwait\_fee"));

highwait\_fee->setGeometry(QRect(180, 120, 161, 16));

tabWidget->addTab(tab\_analytics, QString());

tab\_script = new QWidget();

tab\_script->setObjectName(QStringLiteral("tab\_script"));

witness\_box = new QGroupBox(tab\_script);

witness\_box->setObjectName(QStringLiteral("witness\_box"));

witness\_box->setGeometry(QRect(60, 10, 821, 101));

witness\_text\_edit = new QTextEdit(witness\_box);

witness\_text\_edit->setObjectName(QStringLiteral("witness\_text\_edit"));

witness\_text\_edit->setGeometry(QRect(10, 30, 801, 61));

witness\_script\_box = new QGroupBox(tab\_script);

witness\_script\_box->setObjectName(QStringLiteral("witness\_script\_box"));

witness\_script\_box->setGeometry(QRect(60, 120, 821, 101));

witness\_script\_text\_edit = new QTextEdit(witness\_script\_box);

witness\_script\_text\_edit->setObjectName(QStringLiteral("witness\_script\_text\_edit"));

witness\_script\_text\_edit->setGeometry(QRect(10, 30, 801, 61));

run\_script\_btn = new QPushButton(tab\_script);

run\_script\_btn->setObjectName(QStringLiteral("run\_script\_btn"));

run\_script\_btn->setGeometry(QRect(60, 220, 113, 32));

script\_scrollArea = new QScrollArea(tab\_script);

script\_scrollArea->setObjectName(QStringLiteral("script\_scrollArea"));

script\_scrollArea->setGeometry(QRect(70, 300, 801, 71));

script\_scrollArea->setWidgetResizable(true);

scrollAreaWidgetContents\_2 = new QWidget();

scrollAreaWidgetContents\_2->setObjectName(QStringLiteral("scrollAreaWidgetContents\_2"));

scrollAreaWidgetContents\_2->setGeometry(QRect(0, 0, 799, 69));

script\_scrollArea->setWidget(scrollAreaWidgetContents\_2);

label\_11 = new QLabel(tab\_script);

label\_11->setObjectName(QStringLiteral("label\_11"));

label\_11->setGeometry(QRect(70, 280, 101, 16));

tabWidget->addTab(tab\_script, QString());

app->setCentralWidget(centralWidget);

menuBar = new QMenuBar(app);

menuBar->setObjectName(QStringLiteral("menuBar"));

menuBar->setGeometry(QRect(0, 0, 953, 22));

menuBar->setDefaultUp(true);

menuAtlas = new QMenu(menuBar);

menuAtlas->setObjectName(QStringLiteral("menuAtlas"));

menuHistory = new QMenu(menuBar);

menuHistory->setObjectName(QStringLiteral("menuHistory"));

menuHistory->setAcceptDrops(false);

menuAnalytics = new QMenu(menuBar);

menuAnalytics->setObjectName(QStringLiteral("menuAnalytics"));

menuScript = new QMenu(menuBar);

menuScript->setObjectName(QStringLiteral("menuScript"));

app->setMenuBar(menuBar);

menuBar->addAction(menuAtlas->menuAction());

menuBar->addAction(menuHistory->menuAction());

menuBar->addAction(menuAnalytics->menuAction());

menuBar->addAction(menuScript->menuAction());

menuAtlas->addAction(actionExit);

menuAtlas->addSeparator();

retranslateUi(app);

tabWidget->setCurrentIndex(4);

QMetaObject::connectSlotsByName(app);

} // setupUi

void retranslateUi(QMainWindow \*app)

{

app->setWindowTitle(QApplication::translate("app", "Atlas", nullptr));

actionExit->setText(QApplication::translate("app", "Exit", nullptr));

groupBox->setTitle(QApplication::translate("app", "Overview", nullptr));

label->setText(QApplication::translate("app", "Recieved:", nullptr));

label\_2->setText(QApplication::translate("app", "Sent:", nullptr));

label\_3->setText(QApplication::translate("app", "Balance:", nullptr));

btc\_recieved->setText(QApplication::translate("app", "btc\_recieved", nullptr));

btc\_sent->setText(QApplication::translate("app", "btc\_sent", nullptr));

btc\_balance->setText(QApplication::translate("app", "btc\_balance", nullptr));

groupBox\_2->setTitle(QApplication::translate("app", "Available Payment Address", nullptr));

btc\_address->setText(QApplication::translate("app", "btc\_address", nullptr));

copy\_btc\_address->setText(QApplication::translate("app", "Copy", nullptr));

tabWidget->setTabText(tabWidget->indexOf(tab\_main), QApplication::translate("app", "Main", nullptr));

label\_4->setText(QApplication::translate("app", "Recipient", nullptr));

label\_5->setText(QApplication::translate("app", "Amount", nullptr));

label\_6->setText(QApplication::translate("app", "Fee", nullptr));

send\_tx->setText(QApplication::translate("app", "Send", nullptr));

pushButton\_2->setText(QApplication::translate("app", "Cancel", nullptr));

sat\_byte\_fee->setText(QApplication::translate("app", "0 Satoshis per Byte", nullptr));

label\_7->setText(QApplication::translate("app", "mBTC", nullptr));

tabWidget->setTabText(tabWidget->indexOf(tab\_send), QApplication::translate("app", "Send", nullptr));

tabWidget->setTabText(tabWidget->indexOf(tab\_history), QApplication::translate("app", "History", nullptr));

groupBox\_3->setTitle(QApplication::translate("app", "Fees", nullptr));

label\_8->setText(QApplication::translate("app", "Fastest Fee:", nullptr));

label\_9->setText(QApplication::translate("app", "30-minute Fee:", nullptr));

label\_10->setText(QApplication::translate("app", "1-hour Fee:", nullptr));

fastwait\_fee->setText(QApplication::translate("app", "fastwait\_fee", nullptr));

midwait\_fee->setText(QApplication::translate("app", "midwait\_fee", nullptr));

highwait\_fee->setText(QApplication::translate("app", "highwait\_fee", nullptr));

tabWidget->setTabText(tabWidget->indexOf(tab\_analytics), QApplication::translate("app", "Analytics", nullptr));

witness\_box->setTitle(QApplication::translate("app", "Unlocking Script (Witness)", nullptr));

witness\_script\_box->setTitle(QApplication::translate("app", "Locking Script (Witness Script)", nullptr));

run\_script\_btn->setText(QApplication::translate("app", "Run", nullptr));

label\_11->setText(QApplication::translate("app", "Script Output", nullptr));

tabWidget->setTabText(tabWidget->indexOf(tab\_script), QApplication::translate("app", "Script", nullptr));

menuAtlas->setTitle(QApplication::translate("app", "Atlas", nullptr));

menuHistory->setTitle(QApplication::translate("app", "History", nullptr));

menuAnalytics->setTitle(QApplication::translate("app", "Analytics", nullptr));

menuScript->setTitle(QApplication::translate("app", "Script", nullptr));

} // retranslateUi

};

namespace Ui {

class app: public Ui\_app {};

} // namespace Ui

QT\_END\_NAMESPACE

#endif // UI\_APP\_H

~~~~~~~~~~~~~~~~~~~~~Source code for file moc\_predefs.h~~~~~~~~~~~~~~~~~~~~~

#define OBJC\_NEW\_PROPERTIES 1

#define \_LP64 1

#define \_\_APPLE\_CC\_\_ 6000

#define \_\_APPLE\_\_ 1

#define \_\_ATOMIC\_ACQUIRE 2

#define \_\_ATOMIC\_ACQ\_REL 4

#define \_\_ATOMIC\_CONSUME 1

#define \_\_ATOMIC\_RELAXED 0

#define \_\_ATOMIC\_RELEASE 3

#define \_\_ATOMIC\_SEQ\_CST 5

#define \_\_BIGGEST\_ALIGNMENT\_\_ 16

#define \_\_BLOCKS\_\_ 1

#define \_\_BYTE\_ORDER\_\_ \_\_ORDER\_LITTLE\_ENDIAN\_\_

#define \_\_CHAR16\_TYPE\_\_ unsigned short

#define \_\_CHAR32\_TYPE\_\_ unsigned int

#define \_\_CHAR\_BIT\_\_ 8

#define \_\_CONSTANT\_CFSTRINGS\_\_ 1

#define \_\_DBL\_DECIMAL\_DIG\_\_ 17

#define \_\_DBL\_DENORM\_MIN\_\_ 4.9406564584124654e-324

#define \_\_DBL\_DIG\_\_ 15

#define \_\_DBL\_EPSILON\_\_ 2.2204460492503131e-16

#define \_\_DBL\_HAS\_DENORM\_\_ 1

#define \_\_DBL\_HAS\_INFINITY\_\_ 1

#define \_\_DBL\_HAS\_QUIET\_NAN\_\_ 1

#define \_\_DBL\_MANT\_DIG\_\_ 53

#define \_\_DBL\_MAX\_10\_EXP\_\_ 308

#define \_\_DBL\_MAX\_EXP\_\_ 1024

#define \_\_DBL\_MAX\_\_ 1.7976931348623157e+308

#define \_\_DBL\_MIN\_10\_EXP\_\_ (-307)

#define \_\_DBL\_MIN\_EXP\_\_ (-1021)

#define \_\_DBL\_MIN\_\_ 2.2250738585072014e-308

#define \_\_DECIMAL\_DIG\_\_ \_\_LDBL\_DECIMAL\_DIG\_\_

#define \_\_DEPRECATED 1

#define \_\_DYNAMIC\_\_ 1

#define \_\_ENVIRONMENT\_MAC\_OS\_X\_VERSION\_MIN\_REQUIRED\_\_ 101000

#define \_\_EXCEPTIONS 1

#define \_\_FINITE\_MATH\_ONLY\_\_ 0

#define \_\_FLT\_DECIMAL\_DIG\_\_ 9

#define \_\_FLT\_DENORM\_MIN\_\_ 1.40129846e-45F

#define \_\_FLT\_DIG\_\_ 6

#define \_\_FLT\_EPSILON\_\_ 1.19209290e-7F

#define \_\_FLT\_EVAL\_METHOD\_\_ 0

#define \_\_FLT\_HAS\_DENORM\_\_ 1

#define \_\_FLT\_HAS\_INFINITY\_\_ 1

#define \_\_FLT\_HAS\_QUIET\_NAN\_\_ 1

#define \_\_FLT\_MANT\_DIG\_\_ 24

#define \_\_FLT\_MAX\_10\_EXP\_\_ 38

#define \_\_FLT\_MAX\_EXP\_\_ 128

#define \_\_FLT\_MAX\_\_ 3.40282347e+38F

#define \_\_FLT\_MIN\_10\_EXP\_\_ (-37)

#define \_\_FLT\_MIN\_EXP\_\_ (-125)

#define \_\_FLT\_MIN\_\_ 1.17549435e-38F

#define \_\_FLT\_RADIX\_\_ 2

#define \_\_FXSR\_\_ 1

#define \_\_GCC\_ATOMIC\_BOOL\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_CHAR16\_T\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_CHAR32\_T\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_CHAR\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_INT\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_LLONG\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_LONG\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_POINTER\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_SHORT\_LOCK\_FREE 2

#define \_\_GCC\_ATOMIC\_TEST\_AND\_SET\_TRUEVAL 1

#define \_\_GCC\_ATOMIC\_WCHAR\_T\_LOCK\_FREE 2

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_1 1

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_16 1

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_2 1

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_4 1

#define \_\_GCC\_HAVE\_SYNC\_COMPARE\_AND\_SWAP\_8 1

#define \_\_GLIBCXX\_BITSIZE\_INT\_N\_0 128

#define \_\_GLIBCXX\_TYPE\_INT\_N\_0 \_\_int128

#define \_\_GNUC\_GNU\_INLINE\_\_ 1

#define \_\_GNUC\_MINOR\_\_ 2

#define \_\_GNUC\_PATCHLEVEL\_\_ 1

#define \_\_GNUC\_\_ 4

#define \_\_GNUG\_\_ 4

#define \_\_GXX\_ABI\_VERSION 1002

#define \_\_GXX\_EXPERIMENTAL\_CXX0X\_\_ 1

#define \_\_GXX\_RTTI 1

#define \_\_GXX\_WEAK\_\_ 1

#define \_\_INT16\_C\_SUFFIX\_\_

#define \_\_INT16\_FMTd\_\_ "hd"

#define \_\_INT16\_FMTi\_\_ "hi"

#define \_\_INT16\_MAX\_\_ 32767

#define \_\_INT16\_TYPE\_\_ short

#define \_\_INT32\_C\_SUFFIX\_\_

#define \_\_INT32\_FMTd\_\_ "d"

#define \_\_INT32\_FMTi\_\_ "i"

#define \_\_INT32\_MAX\_\_ 2147483647

#define \_\_INT32\_TYPE\_\_ int

#define \_\_INT64\_C\_SUFFIX\_\_ LL

#define \_\_INT64\_FMTd\_\_ "lld"

#define \_\_INT64\_FMTi\_\_ "lli"

#define \_\_INT64\_MAX\_\_ 9223372036854775807LL

#define \_\_INT64\_TYPE\_\_ long long int

#define \_\_INT8\_C\_SUFFIX\_\_

#define \_\_INT8\_FMTd\_\_ "hhd"

#define \_\_INT8\_FMTi\_\_ "hhi"

#define \_\_INT8\_MAX\_\_ 127

#define \_\_INT8\_TYPE\_\_ signed char

#define \_\_INTMAX\_C\_SUFFIX\_\_ L

#define \_\_INTMAX\_FMTd\_\_ "ld"

#define \_\_INTMAX\_FMTi\_\_ "li"

#define \_\_INTMAX\_MAX\_\_ 9223372036854775807L

#define \_\_INTMAX\_TYPE\_\_ long int

#define \_\_INTMAX\_WIDTH\_\_ 64

#define \_\_INTPTR\_FMTd\_\_ "ld"

#define \_\_INTPTR\_FMTi\_\_ "li"

#define \_\_INTPTR\_MAX\_\_ 9223372036854775807L

#define \_\_INTPTR\_TYPE\_\_ long int

#define \_\_INTPTR\_WIDTH\_\_ 64

#define \_\_INT\_FAST16\_FMTd\_\_ "hd"

#define \_\_INT\_FAST16\_FMTi\_\_ "hi"

#define \_\_INT\_FAST16\_MAX\_\_ 32767

#define \_\_INT\_FAST16\_TYPE\_\_ short

#define \_\_INT\_FAST32\_FMTd\_\_ "d"

#define \_\_INT\_FAST32\_FMTi\_\_ "i"

#define \_\_INT\_FAST32\_MAX\_\_ 2147483647

#define \_\_INT\_FAST32\_TYPE\_\_ int

#define \_\_INT\_FAST64\_FMTd\_\_ "ld"

#define \_\_INT\_FAST64\_FMTi\_\_ "li"

#define \_\_INT\_FAST64\_MAX\_\_ 9223372036854775807L

#define \_\_INT\_FAST64\_TYPE\_\_ long int

#define \_\_INT\_FAST8\_FMTd\_\_ "hhd"

#define \_\_INT\_FAST8\_FMTi\_\_ "hhi"

#define \_\_INT\_FAST8\_MAX\_\_ 127

#define \_\_INT\_FAST8\_TYPE\_\_ signed char

#define \_\_INT\_LEAST16\_FMTd\_\_ "hd"

#define \_\_INT\_LEAST16\_FMTi\_\_ "hi"

#define \_\_INT\_LEAST16\_MAX\_\_ 32767

#define \_\_INT\_LEAST16\_TYPE\_\_ short

#define \_\_INT\_LEAST32\_FMTd\_\_ "d"

#define \_\_INT\_LEAST32\_FMTi\_\_ "i"

#define \_\_INT\_LEAST32\_MAX\_\_ 2147483647

#define \_\_INT\_LEAST32\_TYPE\_\_ int

#define \_\_INT\_LEAST64\_FMTd\_\_ "ld"

#define \_\_INT\_LEAST64\_FMTi\_\_ "li"

#define \_\_INT\_LEAST64\_MAX\_\_ 9223372036854775807L

#define \_\_INT\_LEAST64\_TYPE\_\_ long int

#define \_\_INT\_LEAST8\_FMTd\_\_ "hhd"

#define \_\_INT\_LEAST8\_FMTi\_\_ "hhi"

#define \_\_INT\_LEAST8\_MAX\_\_ 127

#define \_\_INT\_LEAST8\_TYPE\_\_ signed char

#define \_\_INT\_MAX\_\_ 2147483647

#define \_\_LDBL\_DECIMAL\_DIG\_\_ 21

#define \_\_LDBL\_DENORM\_MIN\_\_ 3.64519953188247460253e-4951L

#define \_\_LDBL\_DIG\_\_ 18

#define \_\_LDBL\_EPSILON\_\_ 1.08420217248550443401e-19L

#define \_\_LDBL\_HAS\_DENORM\_\_ 1

#define \_\_LDBL\_HAS\_INFINITY\_\_ 1

#define \_\_LDBL\_HAS\_QUIET\_NAN\_\_ 1

#define \_\_LDBL\_MANT\_DIG\_\_ 64

#define \_\_LDBL\_MAX\_10\_EXP\_\_ 4932

#define \_\_LDBL\_MAX\_EXP\_\_ 16384

#define \_\_LDBL\_MAX\_\_ 1.18973149535723176502e+4932L

#define \_\_LDBL\_MIN\_10\_EXP\_\_ (-4931)

#define \_\_LDBL\_MIN\_EXP\_\_ (-16381)

#define \_\_LDBL\_MIN\_\_ 3.36210314311209350626e-4932L

#define \_\_LITTLE\_ENDIAN\_\_ 1

#define \_\_LONG\_LONG\_MAX\_\_ 9223372036854775807LL

#define \_\_LONG\_MAX\_\_ 9223372036854775807L

#define \_\_LP64\_\_ 1

#define \_\_MACH\_\_ 1

#define \_\_MMX\_\_ 1

#define \_\_NO\_MATH\_INLINES 1

#define \_\_OBJC\_BOOL\_IS\_BOOL 0

#define \_\_OPTIMIZE\_\_ 1

#define \_\_ORDER\_BIG\_ENDIAN\_\_ 4321

#define \_\_ORDER\_LITTLE\_ENDIAN\_\_ 1234

#define \_\_ORDER\_PDP\_ENDIAN\_\_ 3412

#define \_\_PIC\_\_ 2

#define \_\_POINTER\_WIDTH\_\_ 64

#define \_\_PRAGMA\_REDEFINE\_EXTNAME 1

#define \_\_PTRDIFF\_FMTd\_\_ "ld"

#define \_\_PTRDIFF\_FMTi\_\_ "li"

#define \_\_PTRDIFF\_MAX\_\_ 9223372036854775807L

#define \_\_PTRDIFF\_TYPE\_\_ long int

#define \_\_PTRDIFF\_WIDTH\_\_ 64

#define \_\_REGISTER\_PREFIX\_\_

#define \_\_SCHAR\_MAX\_\_ 127

#define \_\_SHRT\_MAX\_\_ 32767

#define \_\_SIG\_ATOMIC\_MAX\_\_ 2147483647

#define \_\_SIG\_ATOMIC\_WIDTH\_\_ 32

#define \_\_SIZEOF\_DOUBLE\_\_ 8

#define \_\_SIZEOF\_FLOAT\_\_ 4

#define \_\_SIZEOF\_INT128\_\_ 16

#define \_\_SIZEOF\_INT\_\_ 4

#define \_\_SIZEOF\_LONG\_DOUBLE\_\_ 16

#define \_\_SIZEOF\_LONG\_LONG\_\_ 8

#define \_\_SIZEOF\_LONG\_\_ 8

#define \_\_SIZEOF\_POINTER\_\_ 8

#define \_\_SIZEOF\_PTRDIFF\_T\_\_ 8

#define \_\_SIZEOF\_SHORT\_\_ 2

#define \_\_SIZEOF\_SIZE\_T\_\_ 8

#define \_\_SIZEOF\_WCHAR\_T\_\_ 4

#define \_\_SIZEOF\_WINT\_T\_\_ 4

#define \_\_SIZE\_FMTX\_\_ "lX"

#define \_\_SIZE\_FMTo\_\_ "lo"

#define \_\_SIZE\_FMTu\_\_ "lu"

#define \_\_SIZE\_FMTx\_\_ "lx"

#define \_\_SIZE\_MAX\_\_ 18446744073709551615UL

#define \_\_SIZE\_TYPE\_\_ long unsigned int

#define \_\_SIZE\_WIDTH\_\_ 64

#define \_\_SSE2\_MATH\_\_ 1

#define \_\_SSE2\_\_ 1

#define \_\_SSE3\_\_ 1

#define \_\_SSE\_MATH\_\_ 1

#define \_\_SSE\_\_ 1

#define \_\_SSP\_\_ 1

#define \_\_SSSE3\_\_ 1

#define \_\_STDCPP\_DEFAULT\_NEW\_ALIGNMENT\_\_ 16UL

#define \_\_STDC\_HOSTED\_\_ 1

#define \_\_STDC\_NO\_THREADS\_\_ 1

#define \_\_STDC\_UTF\_16\_\_ 1

#define \_\_STDC\_UTF\_32\_\_ 1

#define \_\_STDC\_\_ 1

#define \_\_UINT16\_C\_SUFFIX\_\_

#define \_\_UINT16\_FMTX\_\_ "hX"

#define \_\_UINT16\_FMTo\_\_ "ho"

#define \_\_UINT16\_FMTu\_\_ "hu"

#define \_\_UINT16\_FMTx\_\_ "hx"

#define \_\_UINT16\_MAX\_\_ 65535

#define \_\_UINT16\_TYPE\_\_ unsigned short

#define \_\_UINT32\_C\_SUFFIX\_\_ U

#define \_\_UINT32\_FMTX\_\_ "X"

#define \_\_UINT32\_FMTo\_\_ "o"

#define \_\_UINT32\_FMTu\_\_ "u"

#define \_\_UINT32\_FMTx\_\_ "x"

#define \_\_UINT32\_MAX\_\_ 4294967295U

#define \_\_UINT32\_TYPE\_\_ unsigned int

#define \_\_UINT64\_C\_SUFFIX\_\_ ULL

#define \_\_UINT64\_FMTX\_\_ "llX"

#define \_\_UINT64\_FMTo\_\_ "llo"

#define \_\_UINT64\_FMTu\_\_ "llu"

#define \_\_UINT64\_FMTx\_\_ "llx"

#define \_\_UINT64\_MAX\_\_ 18446744073709551615ULL

#define \_\_UINT64\_TYPE\_\_ long long unsigned int

#define \_\_UINT8\_C\_SUFFIX\_\_

#define \_\_UINT8\_FMTX\_\_ "hhX"

#define \_\_UINT8\_FMTo\_\_ "hho"

#define \_\_UINT8\_FMTu\_\_ "hhu"

#define \_\_UINT8\_FMTx\_\_ "hhx"

#define \_\_UINT8\_MAX\_\_ 255

#define \_\_UINT8\_TYPE\_\_ unsigned char

#define \_\_UINTMAX\_C\_SUFFIX\_\_ UL

#define \_\_UINTMAX\_FMTX\_\_ "lX"

#define \_\_UINTMAX\_FMTo\_\_ "lo"

#define \_\_UINTMAX\_FMTu\_\_ "lu"

#define \_\_UINTMAX\_FMTx\_\_ "lx"

#define \_\_UINTMAX\_MAX\_\_ 18446744073709551615UL

#define \_\_UINTMAX\_TYPE\_\_ long unsigned int

#define \_\_UINTMAX\_WIDTH\_\_ 64

#define \_\_UINTPTR\_FMTX\_\_ "lX"

#define \_\_UINTPTR\_FMTo\_\_ "lo"

#define \_\_UINTPTR\_FMTu\_\_ "lu"

#define \_\_UINTPTR\_FMTx\_\_ "lx"

#define \_\_UINTPTR\_MAX\_\_ 18446744073709551615UL

#define \_\_UINTPTR\_TYPE\_\_ long unsigned int

#define \_\_UINTPTR\_WIDTH\_\_ 64

#define \_\_UINT\_FAST16\_FMTX\_\_ "hX"

#define \_\_UINT\_FAST16\_FMTo\_\_ "ho"

#define \_\_UINT\_FAST16\_FMTu\_\_ "hu"

#define \_\_UINT\_FAST16\_FMTx\_\_ "hx"

#define \_\_UINT\_FAST16\_MAX\_\_ 65535

#define \_\_UINT\_FAST16\_TYPE\_\_ unsigned short

#define \_\_UINT\_FAST32\_FMTX\_\_ "X"

#define \_\_UINT\_FAST32\_FMTo\_\_ "o"

#define \_\_UINT\_FAST32\_FMTu\_\_ "u"

#define \_\_UINT\_FAST32\_FMTx\_\_ "x"

#define \_\_UINT\_FAST32\_MAX\_\_ 4294967295U

#define \_\_UINT\_FAST32\_TYPE\_\_ unsigned int

#define \_\_UINT\_FAST64\_FMTX\_\_ "lX"

#define \_\_UINT\_FAST64\_FMTo\_\_ "lo"

#define \_\_UINT\_FAST64\_FMTu\_\_ "lu"

#define \_\_UINT\_FAST64\_FMTx\_\_ "lx"

#define \_\_UINT\_FAST64\_MAX\_\_ 18446744073709551615UL

#define \_\_UINT\_FAST64\_TYPE\_\_ long unsigned int

#define \_\_UINT\_FAST8\_FMTX\_\_ "hhX"

#define \_\_UINT\_FAST8\_FMTo\_\_ "hho"

#define \_\_UINT\_FAST8\_FMTu\_\_ "hhu"

#define \_\_UINT\_FAST8\_FMTx\_\_ "hhx"

#define \_\_UINT\_FAST8\_MAX\_\_ 255

#define \_\_UINT\_FAST8\_TYPE\_\_ unsigned char

#define \_\_UINT\_LEAST16\_FMTX\_\_ "hX"

#define \_\_UINT\_LEAST16\_FMTo\_\_ "ho"

#define \_\_UINT\_LEAST16\_FMTu\_\_ "hu"

#define \_\_UINT\_LEAST16\_FMTx\_\_ "hx"

#define \_\_UINT\_LEAST16\_MAX\_\_ 65535

#define \_\_UINT\_LEAST16\_TYPE\_\_ unsigned short

#define \_\_UINT\_LEAST32\_FMTX\_\_ "X"

#define \_\_UINT\_LEAST32\_FMTo\_\_ "o"

#define \_\_UINT\_LEAST32\_FMTu\_\_ "u"

#define \_\_UINT\_LEAST32\_FMTx\_\_ "x"

#define \_\_UINT\_LEAST32\_MAX\_\_ 4294967295U

#define \_\_UINT\_LEAST32\_TYPE\_\_ unsigned int

#define \_\_UINT\_LEAST64\_FMTX\_\_ "lX"

#define \_\_UINT\_LEAST64\_FMTo\_\_ "lo"

#define \_\_UINT\_LEAST64\_FMTu\_\_ "lu"

#define \_\_UINT\_LEAST64\_FMTx\_\_ "lx"

#define \_\_UINT\_LEAST64\_MAX\_\_ 18446744073709551615UL

#define \_\_UINT\_LEAST64\_TYPE\_\_ long unsigned int

#define \_\_UINT\_LEAST8\_FMTX\_\_ "hhX"

#define \_\_UINT\_LEAST8\_FMTo\_\_ "hho"

#define \_\_UINT\_LEAST8\_FMTu\_\_ "hhu"

#define \_\_UINT\_LEAST8\_FMTx\_\_ "hhx"

#define \_\_UINT\_LEAST8\_MAX\_\_ 255

#define \_\_UINT\_LEAST8\_TYPE\_\_ unsigned char

#define \_\_USER\_LABEL\_PREFIX\_\_ \_

#define \_\_VERSION\_\_ "4.2.1 Compatible Apple LLVM 9.0.0 (clang-900.0.39.2)"

#define \_\_WCHAR\_MAX\_\_ 2147483647

#define \_\_WCHAR\_TYPE\_\_ int

#define \_\_WCHAR\_WIDTH\_\_ 32

#define \_\_WINT\_TYPE\_\_ int

#define \_\_WINT\_WIDTH\_\_ 32

#define \_\_amd64 1

#define \_\_amd64\_\_ 1

#define \_\_apple\_build\_version\_\_ 9000039

#define \_\_block \_\_attribute\_\_((\_\_blocks\_\_(byref)))

#define \_\_clang\_\_ 1

#define \_\_clang\_major\_\_ 9

#define \_\_clang\_minor\_\_ 0

#define \_\_clang\_patchlevel\_\_ 0

#define \_\_clang\_version\_\_ "9.0.0 (clang-900.0.39.2)"

#define \_\_core2 1

#define \_\_core2\_\_ 1

#define \_\_cplusplus 201103L

#define \_\_cpp\_alias\_templates 200704

#define \_\_cpp\_attributes 200809

#define \_\_cpp\_constexpr 200704

#define \_\_cpp\_decltype 200707

#define \_\_cpp\_delegating\_constructors 200604

#define \_\_cpp\_exceptions 199711

#define \_\_cpp\_inheriting\_constructors 201511

#define \_\_cpp\_initializer\_lists 200806

#define \_\_cpp\_lambdas 200907

#define \_\_cpp\_nsdmi 200809

#define \_\_cpp\_range\_based\_for 200907

#define \_\_cpp\_raw\_strings 200710

#define \_\_cpp\_ref\_qualifiers 200710

#define \_\_cpp\_rtti 199711

#define \_\_cpp\_rvalue\_references 200610

#define \_\_cpp\_static\_assert 200410

#define \_\_cpp\_unicode\_characters 200704

#define \_\_cpp\_unicode\_literals 200710

#define \_\_cpp\_user\_defined\_literals 200809

#define \_\_cpp\_variadic\_templates 200704

#define \_\_llvm\_\_ 1

#define \_\_nonnull \_Nonnull

#define \_\_null\_unspecified \_Null\_unspecified

#define \_\_nullable \_Nullable

#define \_\_pic\_\_ 2

#define \_\_private\_extern\_\_ extern

#define \_\_strong

#define \_\_tune\_core2\_\_ 1

#define \_\_unsafe\_unretained

#define \_\_weak \_\_attribute\_\_((objc\_gc(weak)))

#define \_\_x86\_64 1

#define \_\_x86\_64\_\_ 1

~~~~~~~~~~~~~~~~~~~~~Source code for file moc\_restore\_wallet.cpp~~~~~~~~~~~~~~~~~~~~~

/\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*

\*\* Meta object code from reading C++ file 'restore\_wallet.h'

\*\*

\*\* Created by: The Qt Meta Object Compiler version 67 (Qt 5.10.1)

\*\*

\*\* WARNING! All changes made in this file will be lost!

\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/

#include "../Atlas/restore\_wallet.h"

#include <QtCore/qbytearray.h>

#include <QtCore/qmetatype.h>

#if !defined(Q\_MOC\_OUTPUT\_REVISION)

#error "The header file 'restore\_wallet.h' doesn't include <QObject>."

#elif Q\_MOC\_OUTPUT\_REVISION != 67

#error "This file was generated using the moc from 5.10.1. It"

#error "cannot be used with the include files from this version of Qt."

#error "(The moc has changed too much.)"

#endif

QT\_BEGIN\_MOC\_NAMESPACE

QT\_WARNING\_PUSH

QT\_WARNING\_DISABLE\_DEPRECATED

struct qt\_meta\_stringdata\_restore\_wallet\_t {

QByteArrayData data[1];

char stringdata0[15];

};

#define QT\_MOC\_LITERAL(idx, ofs, len) \

Q\_STATIC\_BYTE\_ARRAY\_DATA\_HEADER\_INITIALIZER\_WITH\_OFFSET(len, \

qptrdiff(offsetof(qt\_meta\_stringdata\_restore\_wallet\_t, stringdata0) + ofs \

- idx \* sizeof(QByteArrayData)) \

)

static const qt\_meta\_stringdata\_restore\_wallet\_t qt\_meta\_stringdata\_restore\_wallet = {

{

QT\_MOC\_LITERAL(0, 0, 14) // "restore\_wallet"

},

"restore\_wallet"

};

#undef QT\_MOC\_LITERAL

static const uint qt\_meta\_data\_restore\_wallet[] = {

// content:

7, // revision

0, // classname

0, 0, // classinfo

0, 0, // methods

0, 0, // properties

0, 0, // enums/sets

0, 0, // constructors

0, // flags

0, // signalCount

0 // eod

};

void restore\_wallet::qt\_static\_metacall(QObject \*\_o, QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

Q\_UNUSED(\_o);

Q\_UNUSED(\_id);

Q\_UNUSED(\_c);

Q\_UNUSED(\_a);

}

QT\_INIT\_METAOBJECT const QMetaObject restore\_wallet::staticMetaObject = {

{ &QDialog::staticMetaObject, qt\_meta\_stringdata\_restore\_wallet.data,

qt\_meta\_data\_restore\_wallet, qt\_static\_metacall, nullptr, nullptr}

};

const QMetaObject \*restore\_wallet::metaObject() const

{

return QObject::d\_ptr->metaObject ? QObject::d\_ptr->dynamicMetaObject() : &staticMetaObject;

}

void \*restore\_wallet::qt\_metacast(const char \*\_clname)

{

if (!\_clname) return nullptr;

if (!strcmp(\_clname, qt\_meta\_stringdata\_restore\_wallet.stringdata0))

return static\_cast<void\*>(this);

return QDialog::qt\_metacast(\_clname);

}

int restore\_wallet::qt\_metacall(QMetaObject::Call \_c, int \_id, void \*\*\_a)

{

\_id = QDialog::qt\_metacall(\_c, \_id, \_a);

return \_id;

}

QT\_WARNING\_POP

QT\_END\_MOC\_NAMESPACE

~~~~~~~~~~~~~~~~~~~~~Source code for file operation.cpp~~~~~~~~~~~~~~~~~~~~~

/\*\*

\* @brief Implementation of Operation class.

\*

\* @file operation.cpp

\* @author Philip Glazman

\* @date 5/3/18

\*/

#include "../wallet/stdafx.h"

/\*\*

\* @brief Construct a new Operation:: Operation object

\*

\* @author Philip Glazman

\* @date 5/3/18

\*/

Operation::Operation()

{

// Operation Codes are loaded into hash map for efficient lookup.

// Insert operation codes for stack manipulation.

m\_op\_code\_map.emplace("OP\_DROP",this->OP\_DROP);

m\_op\_code\_map.emplace("OP\_DUP",this->OP\_DUP);

m\_op\_code\_map.emplace("OP\_DEPTH",this->OP\_DEPTH);

// Insert operation codes for binary arithmetic.

m\_op\_code\_map.emplace("OP\_EQUAL",this->OP\_EQUAL);

// Insert operation codes for arithmetic.

m\_op\_code\_map.emplace("OP\_1ADD",this->OP\_1ADD);

m\_op\_code\_map.emplace("OP\_1SUB",this->OP\_1SUB);

m\_op\_code\_map.emplace("OP\_NEGATE",this->OP\_NEGATE);

m\_op\_code\_map.emplace("OP\_ABS",this->OP\_ABS);

m\_op\_code\_map.emplace("OP\_ADD",this->OP\_ADD);

m\_op\_code\_map.emplace("OP\_SUB",this->OP\_SUB);

m\_op\_code\_map.emplace("OP\_NUMEQUAL",this->OP\_NUMEQUAL);

m\_op\_code\_map.emplace("OP\_NUMNOTEQUAL",this->OP\_NUMNOTEQUAL);

m\_op\_code\_map.emplace("OP\_LESSTHAN",this->OP\_LESSTHAN);

m\_op\_code\_map.emplace("OP\_GREATERTHAN",this->OP\_GREATERTHAN);

m\_op\_code\_map.emplace("OP\_LESSTHANOREQUAL",this->OP\_LESSTHANOREQUAL);

m\_op\_code\_map.emplace("OP\_GREATERTHANOREQUAL",this->OP\_GREATERTHANOREQUAL);

m\_op\_code\_map.emplace("OP\_MIN",this->OP\_MIN);

m\_op\_code\_map.emplace("OP\_MAX",this->OP\_MAX);

m\_op\_code\_map.emplace("OP\_WITHIN",this->OP\_WITHIN);

// Insert operation codes for cryptography.

m\_op\_code\_map.emplace("OP\_RIPEMD160",this->OP\_RIPEMD160);

m\_op\_code\_map.emplace("OP\_SHA1",this->OP\_SHA1);

m\_op\_code\_map.emplace("OP\_SHA256",this->OP\_SHA256);

m\_op\_code\_map.emplace("OP\_HASH160",this->OP\_HASH160);

m\_op\_code\_map.emplace("OP\_HASH256",this->OP\_HASH256);

};

/\*\*

\* @brief Operator for pushing value "1" onto the stack.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_1(std::stack<std::string>& a\_stack)

{

a\_stack.push("1");

return a\_stack;

};

/\*\*

\* @brief Operator for pushing an emptry array onto the stack.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

OP\_0(std::stack<std::string>& a\_stack)

{

a\_stack.push({});

return a\_stack;

};

/\*\*

\* @brief Operator for pushing empty array onto the stack.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

OP\_FALSE(std::stack<std::string>& a\_stack)

{

a\_stack.push({});

return a\_stack;

};

/\*\*

\* @brief Operator for popping two top items, adding them, and pushing result onto stack.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

Operation::stack

Operation::OP\_ADD(Operation::stack a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

a\_stack.push(std::to\_string(x+y));

return a\_stack;

};

/\*\*

\* @brief Flips the sign of the top item.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_NEGATE(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

x \*= -1;

a\_stack.push(std::to\_string(x));

return a\_stack;

};

/\*\*

\* @brief Operator for popping two top items, subtracting them, and pushing result onto stack.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_SUB(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

// Subtract first from second

a\_stack.push(std::to\_string(y-x));

return a\_stack;

};

/\*\*

\* @brief Operator that returns true if top two items are equal numbers.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_NUMEQUAL(std::stack<std::string>& a\_stack)

{

std::string x = a\_stack.top();

a\_stack.pop();

std::string y = a\_stack.top();

a\_stack.pop();

if(x==y)

{

a\_stack.push("TRUE");

}

else

{

a\_stack.push("FALSE");

}

return a\_stack;

};

/\*\*

\* @brief Operator that returns true if top two items are not equal numbers.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_NUMNOTEQUAL(std::stack<std::string>& a\_stack)

{

std::string x = a\_stack.top();

a\_stack.pop();

std::string y = a\_stack.top();

a\_stack.pop();

if(x!=y)

{

a\_stack.push("TRUE");

}

else

{

a\_stack.push("FALSE");

}

return a\_stack;

};

/\*\*

\* @brief Operator for pushing 1 if top two items are equal, push 0 if otherwise.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_EQUAL(std::stack<std::string>& a\_stack)

{

std::string x = a\_stack.top();

a\_stack.pop();

std::string y = a\_stack.top();

a\_stack.pop();

if(x==y)

{

a\_stack.push("1");

}

else

{

a\_stack.push("0");

}

return a\_stack;

};

/\*\*

\* @brief Operator for incrementing top value of stack.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_1ADD(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

x++;

a\_stack.push(std::to\_string(x));

return a\_stack;

}

/\*\*

\* @brief Operator for decrementing top value of stack.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_1SUB(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

x--;

a\_stack.push(std::to\_string(x));

return a\_stack;

}

/\*\*

\* @brief Operator for changing the sign of the top item to positive.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_ABS(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

x = std::abs(x);

a\_stack.push(std::to\_string(x));

return a\_stack;

}

/\*\*

\* @brief Operator that returns true if second item is less than top item.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_LESSTHAN(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

if( y < x)

{

a\_stack.push("TRUE");

}

else

{

a\_stack.push("FALSE");

}

return a\_stack;

};

/\*\*

\* @brief Operator that returns true if second item is greater than top item.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_GREATERTHAN(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

if( y > x)

{

a\_stack.push("TRUE");

}

else

{

a\_stack.push("FALSE");

}

return a\_stack;

};

/\*\*

\* @brief Operator that returns true if second item is less than or equal to top item.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_LESSTHANOREQUAL(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

if( y <= x)

{

a\_stack.push("TRUE");

}

else

{

a\_stack.push("FALSE");

}

return a\_stack;

};

/\*\*

\* @brief Operator that returns true if second item is greater than or equal to top item.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_GREATERTHANOREQUAL(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

if( y >= x)

{

a\_stack.push("TRUE");

}

else

{

a\_stack.push("FALSE");

}

return a\_stack;

};

/\*\*

\* @brief Operator that pushes onto the stack the minimum item of the top two items.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_MIN(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

if( x < y)

{

a\_stack.push(std::to\_string(x));

}

else

{

a\_stack.push(std::to\_string(y));

}

return a\_stack;

};

/\*\*

\* @brief Operator that pushes onto the stack the maximum item of the top two items.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_MAX(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

if( x > y)

{

a\_stack.push(std::to\_string(x));

}

else

{

a\_stack.push(std::to\_string(y));

}

return a\_stack;

};

/\*\*

\* @brief

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_WITHIN(std::stack<std::string>& a\_stack)

{

int x = std::stoi(a\_stack.top());

a\_stack.pop();

int y = std::stoi(a\_stack.top());

a\_stack.pop();

int z = std::stoi(a\_stack.top());

a\_stack.pop();

if( z >= y && z < x)

{

a\_stack.push("TRUE");

}

else

{

a\_stack.push("FALSE");

}

return a\_stack;

};

/\*\*

\* @brief Operator for popping the top item off the stack

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_DROP(std::stack<std::string>& a\_stack)

{

if(!a\_stack.empty())

{

a\_stack.pop();

};

return a\_stack;

};

/\*\*

\* @brief Duplicates the tope item in the stack.

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_DUP(std::stack<std::string>& a\_stack)

{

a\_stack.push(a\_stack.top());

return a\_stack;

};

/\*\*

\* @brief Operator for counting the items on the stack and pushing the result onto the stack

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_DEPTH(std::stack<std::string>& a\_stack)

{

std::string count\_stack = std::to\_string(a\_stack.size());

a\_stack.push(count\_stack);

return a\_stack;

};

/\*\*

\* @brief Operator for pushing RIPEMD160 hash of the top item onto the stack

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_RIPEMD160(std::stack<std::string>& a\_stack)

{

std::string hashed\_string = hash\_RIPEMD160(a\_stack.top());

a\_stack.pop();

a\_stack.push(hashed\_string);

};

/\*\*

\* @brief Operator for pushing SHA1 hash of the top item onto the stack

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_SHA1(std::stack<std::string>& a\_stack)

{

std::string hashed\_string = hash\_SHA1(a\_stack.top());

a\_stack.pop();

a\_stack.push(hashed\_string);

};

/\*\*

\* @brief Operator for pushing SHA256 hash of the top item onto the stack

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_SHA256(std::stack<std::string>& a\_stack)

{

std::string hashed\_string = hash\_SHA256(a\_stack.top());

a\_stack.pop();

a\_stack.push(hashed\_string);

};

/\*\*

\* @brief Operator for pushing RIPEMD160(SHA256(n)) hash of the top item onto the stack

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_HASH160(std::stack<std::string>& a\_stack)

{

// SHA256 hash string

std::string sha256\_hashed\_string = hash\_SHA256(a\_stack.top());

// Double hash with RIPMD160

std::string hashed\_string = hash\_RIPEMD160(sha256\_hashed\_string);

a\_stack.pop();

a\_stack.push(hashed\_string);

};

/\*\*

\* @brief Operator for pushing SHA256(SHA256(n)) hash of the top item onto the stack

\*

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::OP\_HASH256(std::stack<std::string>& a\_stack)

{

// SHA256 hash string

std::string sha256\_hashed\_string = hash\_SHA256(a\_stack.top());

// Double SHA256 hash

std::string hashed\_string = hash\_SHA256(sha256\_hashed\_string);

a\_stack.pop();

a\_stack.push(hashed\_string);

};

/\*\*

\* @brief Calls appropriate operation codes and changes stack.

\*

\* @param a\_code

\* @param a\_stack

\* @return std::stack<std::string>&

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::stack<std::string>&

Operation::call\_operation(std::string a\_code,std::stack<std::string> &a\_stack)

{

if(a\_code.length() == 1)

{

try

{

// Change to integer.

int n = std::stoi(a\_code);

// 1- 75

if(n > 0 && n < 76)

{

a\_stack.push(std::to\_string(n));

return a\_stack;

};

}

catch(std::exception e)

{

std::cout << e.what() << std::endl;

}

};

// Iterator for op code hash map.

std::unordered\_map<std::string,func>::iterator iter;

iter = m\_op\_code\_map.find(a\_code);

if(iter != m\_op\_code\_map.end())

{

std::cout << "Operation Code Found" << std::endl;

a\_stack = (\*iter->second)(a\_stack);

}

else

{

std::cout << "Operation Code NOT Found" << std::endl;

}

return a\_stack;

};

/\*\*

\* @brief Applies SHA256 hash function on a string.

\*

\* @return std::string

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::string

Operation::hash\_SHA256(std::string a\_string)

{

// Digest

unsigned char digest[SHA256\_DIGEST\_LENGTH];

char str[a\_string.length()];

strncpy(str,a\_string.c\_str(),sizeof(a\_string));

SHA256((unsigned char\*)&str,strlen(str),(unsigned char\*)&digest);

char mdString[SHA256\_DIGEST\_LENGTH\*2+1];

for(int i = 0; i < SHA256\_DIGEST\_LENGTH; i++)

{

sprintf(&mdString[i\*2], "%02x", (unsigned int)digest[i]);

};

return std::string(mdString);

};

/\*\*

\* @brief Applies RIPEMD160 hash function a string.

\*

\* @return std::string

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::string

Operation::hash\_RIPEMD160(std::string a\_string)

{

unsigned char digest[RIPEMD160\_DIGEST\_LENGTH];

char str[a\_string.length()];

strncpy(str,a\_string.c\_str(),sizeof(a\_string));

RIPEMD160((unsigned char\*)&str,strlen(str),(unsigned char\*)&digest);

char mdString[RIPEMD160\_DIGEST\_LENGTH\*2+1];

for(int i = 0; i < RIPEMD160\_DIGEST\_LENGTH; i++)

{

sprintf(&mdString[i\*2], "%02x", (unsigned int)digest[i]);

}

return std::string(mdString);

};

/\*\*

\* @brief Applies SHA1 hash function on string.

\*

\* @return std::string

\*

\* @author Philip Glazman

\* @date 5/2/18

\*/

std::string

Operation::hash\_SHA1(std::string a\_string)

{

unsigned char digest[SHA\_DIGEST\_LENGTH];

char str[a\_string.length()];

strncpy(str,a\_string.c\_str(),sizeof(a\_string));

SHA1((unsigned char\*)&str,strlen(str),(unsigned char\*)&digest);

char mdString[SHA\_DIGEST\_LENGTH\*2+1];

for(int i = 0; i < SHA\_DIGEST\_LENGTH; i++)

{

sprintf(&mdString[i\*2], "%02x", (unsigned int)digest[i]);

}

return std::string(mdString);

};

~~~~~~~~~~~~~~~~~~~~~Source code for file script.cpp~~~~~~~~~~~~~~~~~~~~~

/\*\*

\* @brief Implementation of the Script class.

\*

\* @file script.cpp

\* @author Philip Glazman

\* @date 5/3/18

\*/

#include "../wallet/stdafx.h"

/\*\*

\* @brief Construct a new script::script object

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

Script::Script()

{

// Configure current consensus rules for the stack to comply with.

m\_fork\_rules = bc::machine::rule\_fork::all\_rules;

m\_operation = new Operation::Operation();

};

/\*\*

\* @brief Destroy the script::script object

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

Script::~Script()

{

delete m\_operation;

};

/\*\*

\* @brief Evaluates the script on the current execution stack and returns if script is valid.

\*

\* @author Philip Glazman

\* @date 5/3/18

\*/

bool

Script::is\_valid()

{

if(m\_execution\_stack.size()==1 and (m\_execution\_stack.top()=="1" || m\_execution\_stack.top()=="True"))

{

return true;

}

else

{

return false;

}

};

/\*\*

\* @brief

\*

\* @param witness

\* @param witness\_script

\* @return true

\* @return false

\*

\* @author Philip Glazman

\* @date 4/30/18

\*/

bool

Script::build\_script(std::string a\_witness, std::string a\_witness\_script)

{

std::string script = a\_witness +" "+ a\_witness\_script;

// Use istringstream class to parse witness and witness script.

std::istringstream execution\_item (script);

// Operater/Operand at specific point in script.

std::string execution\_pointer;

while(execution\_item)

{

execution\_pointer.clear();

execution\_item >> execution\_pointer;

if( execution\_pointer != "" )

{

// Push the witness onto the execution stack.

std::cout << execution\_pointer << std::endl;

m\_operation->call\_operation(execution\_pointer,m\_execution\_stack);

}

};

};

/\*\*

\* @brief Clears the current execution stack.

\*

\* @return true

\* @return false

\*

\* @author Philip Glazman

\* @date 5/3/18

\*/

bool

Script::clear\_script()

{

while(!m\_execution\_stack.empty())

{

m\_execution\_stack.pop();

}

return true;

};

~~~~~~~~~~~~~~~~~~~~~Source code for file debug.cpp~~~~~~~~~~~~~~~~~~~~~

#include "../wallet/stdafx.h"

int main()

{

/\*std::stack<std::string> my\_stack;

Operation operators;

operators.call\_operation("1",my\_stack);

operators.call\_operation("2",my\_stack);

operators.call\_operation("OP\_ADD",my\_stack);

operators.call\_operation("3",my\_stack);

operators.call\_operation("OP\_EQUAL",my\_stack);

while(!my\_stack.empty())

{

std::cout << my\_stack.top() << std::endl;

my\_stack.pop();

}\*/

Script my\_script;

std::string witness = "1 2";

std::string witness\_script = "OP\_ADD 4 OP\_EQUAL";

my\_script.build\_script(witness, witness\_script);

std::cout << my\_script.is\_valid() << std::endl;

}

~~~~~~~~~~~~~~~~~~~~~Source code for file stdafx.h~~~~~~~~~~~~~~~~~~~~~

// stdafx.h : include file for standard system include files

/\*

TO

compile as g++ -c stdafx.h -o stdafx.h.gch

g++ -c stdafx.h -o stdafx.h.gch -std=c++11 -lboostsystem -lbitcoin

\*/

#pragma once

// TODO: add additional headers to the program

// Libbitcoin

#include <bitcoin/bitcoin.hpp>

#include <bitcoin/client.hpp>

// STL

#include <string.h>

#include <iostream>

#include <cstdint>

#include <string>

#include <vector>

#include <iomanip>

#include <random>

#include <unordered\_map>

#include <sstream>

#include <stack>

// Utilities

#include <curl/curl.h>

#include <json/json.h>

// Boost Libraries

#include <boost/property\_tree/ptree.hpp>

#include <boost/property\_tree/json\_parser.hpp>

#include <boost/asio.hpp>

// Atlas Headers

#include "error.hpp"

#include "../network/network.hpp"

#include "../script/script.hpp"

#include "../script/operation.hpp"

#include "Wallet.hpp"

#include "utxo.hpp"

#include "transaction.hpp"

// Crypto Libraries

#include <openssl/ripemd.h>

#include <openssl/sha.h>

~~~~~~~~~~~~~~~~~~~~~Source code for file error.cpp~~~~~~~~~~~~~~~~~~~~~

/\*

\* Implementation of the Error class.

\*/

#include "stdafx.h"

// Initializes error reports.

std::queue<std::string> Error::m\_ErrorMsgs;

/\*\*/

/\*

Errors::InitErrorReporting()

NAME

Errors::InitErrorReporting()

SYNOPSIS

void Errors::InitErrorReporting()

DESCRIPTION

This function empties the error queue in order to remove any junk.

RETURNS

Returns nothing

AUTHOR

Philip Glazman

DATE

1/8/2018

\*/

/\*\*/

void

Error::InitErrorReporting()

{

while (!m\_ErrorMsgs.empty()) m\_ErrorMsgs.pop();

}

/\*\*/

/\*

Errors::RecordError(string &a\_emsg)

NAME

Errors::RecordError(string &a\_emsg)

SYNOPSIS

void Errors::RecordError(string &a\_emsg)

a\_emsg --> Error message to push to the queue.

DESCRIPTION

This function pushes a string error message to the queue.

RETURNS

Returns nothing.

AUTHOR

Philip Glazman

DATE

1/8/2018

\*/

/\*\*/

void

Error::RecordError(std::string a\_emsg)

{

m\_ErrorMsgs.push(a\_emsg);

}

/\*\*/

/\*

Errors::DisplayErrors()

NAME

Errors::DisplayErrors()

SYNOPSIS

void Errors::DisplayErrors()

DESCRIPTION

This function outputs any error messages in the queue.

RETURNS

Returns nothing.

AUTHOR

Philip Glazman

DATE

1/8/2018

\*/

/\*\*/

void

Error::DisplayErrors()

{

// While there are any error messages, print them to the screen.

while (!m\_ErrorMsgs.empty())

{

std::cout << std::setw(15) << std::right << m\_ErrorMsgs.front() << std::endl;

m\_ErrorMsgs.pop();

}

}

~~~~~~~~~~~~~~~~~~~~~Source code for file atlas.cpp~~~~~~~~~~~~~~~~~~~~~

/\*

\* Main program for Atlas.

g++ -std=c++11 -o atlaswallet atlas.cpp wallet.cpp error.cpp transaction.cpp ../network/network.cpp utxo.cpp $(pkg-config --cflags libbitcoin --libs libbitcoin libbitcoin-client libcurl jsoncpp)

\*/

#include "stdafx.h"

int

main(int argc, char \* argv[])

{

// Load wallet.

std::vector< std::string > wordList = {"scatter", "found", "issue", "friend", "front", "glare", "blanket", "mother", "frequent", "acid", "shaft", "loud"};

// Wallet object.

Wallet wallet(wordList);

// Reveal keys.

// wallet.showKeys();

// Transactions object;

// Transaction transactions;

// // Check balance.

// // int addressIndex = 1;

// // while(true)

// // {

// // if(transactions.calculateBalance(wallet.getAddress(addressIndex)))

// // {

// // addressIndex++;

// // }

// // else

// // {

// // break;

// // }

// // };

// // wallet.set\_address\_index\_to\_last\_unused\_address();

// std::cout<<wallet.getBalance()<<std::endl;

// std::cout<<wallet.get\_balance\_as\_string()<<std::endl;

// std::cout<<wallet.getAddress(1).is\_address()<<std::endl;

// std::cout << addressIndex << std::endl;

// std::cout << transactions.getBalance() << std::endl;

// bc::wallet::payment\_address addy = wallet.getAddress(1);

// bc::wallet::payment\_address destinationAddy = wallet.getAddress(3);

// bc::data\_chunk publicKey = bc::to\_chunk(wallet.childPublicKey(1).point());

// wallet.build\_P2PKH("mmUbEcLMoJsaT6Uy3ZBkvF5i1AJ5xgmZpG",1000000);

// transactions.P2PKH(destinationAddy, 1000000);

// // Fees

// Network net;

// net.refreshFeeRecommendations();

};

~~~~~~~~~~~~~~~~~~~~~Source code for file utxo.cpp~~~~~~~~~~~~~~~~~~~~~

#include "stdafx.h"

/\*\*

\* @brief Construct a new utxo::utxo object

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

utxo::utxo()

{

m\_tx\_output = new std::vector < std::tuple <m\_satoshis, m\_utxo\_hash, m\_address> >;

};

/\*\*

\* @brief Destroy the utxo::utxo object

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

utxo::~utxo()

{

delete m\_tx\_output;

};

/\*\*

\* @brief Adds a transaction to the utxo map.

\*

\* @param a\_satoshis

\* @param a\_utxo\_hash

\* @param a\_address

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void utxo::add\_transaction(unsigned long long a\_satoshis, bc::hash\_digest a\_utxo\_hash, bc::wallet::payment\_address a\_address) const

{

m\_tx\_output -> push\_back( std::make\_tuple(a\_satoshis, a\_utxo\_hash, a\_address));

}

/\*\*

\* @brief Returns value of a transaction hash.

\*

\* @param a\_utxo\_hash

\* @return unsigned long long

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

unsigned long long utxo::get\_value(bc::hash\_digest a\_utxo\_hash)

{

//@TODO - update

return 1;

};

// bool compare\_utxo(const utxo\_tuple& lhs, const utxo\_tuple& rhs)

// {

// return std::get<0>(lhs) < std::get<0>(rhs);

// };

void utxo::show\_available\_utxo()

{

for(const auto&tx : \*m\_tx\_output)

{

std::cout << "Payment Address: " << std::get<2>(tx) << " Value: " << std::get<0>(tx) << " UTXO Hash: " << bc::encode\_hash(std::get<1>(tx)) << std::endl;

};

};

// finds the minimum utxo to satisfy need

// returns stack of utxos which conntains tuple of payment address, utxo hash, and value

utxo::utxo\_data utxo::find\_utxo(unsigned long long a\_satoshis)

{

// Used to sum each utxo value

unsigned long long value = 0;

utxo\_data utxo\_to\_return;

// Sort vector in ascending order (min to max) according to utxo value.

std::sort(m\_tx\_output->begin(), m\_tx\_output->end(),compare\_utxo());

show\_available\_utxo();

for(const auto&tx : \*m\_tx\_output)

{

if(value > a\_satoshis)

{

break;

}

utxo\_to\_return.push\_back(tx);

value += std::get<0>(tx);

};

return utxo\_to\_return;

};

// get utxo based on what to spend - get lowest

~~~~~~~~~~~~~~~~~~~~~Source code for file Wallet.cpp~~~~~~~~~~~~~~~~~~~~~

#include "stdafx.h"

/\*\*

\* @brief Creates new wallet using user entropy (256 bits).

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

Wallet::Wallet()

{

bc::wallet::word\_list mnemonicSeed = generateMnemonicCode();

m\_seed = bc::to\_chunk(bc::wallet::decode\_mnemonic(mnemonicSeed));

m\_mnemonic = mnemonicSeed;

// Master 256-bit Private Key.

m\_masterPrivateKey = bc::wallet::hd\_private(m\_seed,bc::wallet::hd\_private::testnet);

// Master 264-bit Public Key.

m\_masterPublicKey = m\_masterPrivateKey.to\_public();

// Transactions object.

transactions = new Transaction();

m\_address\_index=1;

set\_address\_index\_to\_last\_unused\_address();

}

/\*\*

\* @brief Creates new wallet by import 12 word phrase.

\*

\* @param a\_mnemonicSeed, bc::wallet::word\_list. List of 12 word seed phrase.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

Wallet::Wallet(const bc::wallet::word\_list a\_mnemonicSeed)

{

// 512 bit seed is derived from mnemonic bits.

m\_seed = bc::to\_chunk(bc::wallet::decode\_mnemonic(a\_mnemonicSeed));

m\_mnemonic = a\_mnemonicSeed;

// Master 256-bit Private Key.

m\_masterPrivateKey = bc::wallet::hd\_private(m\_seed,bc::wallet::hd\_private::testnet);

// Master 264-bit Public Key.

m\_masterPublicKey = m\_masterPrivateKey.to\_public();

// Transactions object.

transactions = new Transaction();

m\_address\_index=1;

set\_address\_index\_to\_last\_unused\_address();

}

/\*\*

\* @brief Generates mnemonic bits using user machine's entropy. BIP-39 Standard.

\*

\* @return bc::wallet::word\_list. List of 12 words representing seed of wallet.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::wallet::word\_list

Wallet::generateMnemonicCode()

{

// Store 128 bits for entropy.

m\_entropy = new std::vector<std::uint8\_t>(16);

// Entropy is generated using local machine.

bc::pseudo\_random\_fill(\*m\_entropy);

// Entropy is included in bits to generate mnemonic words.

bc::wallet::word\_list mnemonicSeed = bc::wallet::create\_mnemonic(\*m\_entropy);

delete m\_entropy;

return mnemonicSeed;

// Create 512-bit seed using mnemonic code wirds and a\_passphrase as Salt.

// TODO - add ICU to library dependency to make it work with passphrase

};

/\*\*

\* @brief Selector for child private key at index n of keychain.

\*

\* @param a\_index, integer.

\* @return bc::wallet::hd\_private

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::wallet::hd\_private

Wallet::childPrivateKey(int a\_index)

{

return m\_masterPrivateKey.derive\_private(a\_index);

}

/\*\*

\* @brief Selector for child public key at index n of keychain.

\*

\* @param a\_index, integer.

\* @return bc::wallet::hd\_public

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::wallet::hd\_public

Wallet::childPublicKey(int a\_index)

{

return m\_masterPublicKey.derive\_public(a\_index);

}

/\*\*

\* @brief Return the Bitcoin Address (Base58 encoded address) at index n of keychain.

\*

\* @param a\_index, integer.

\* @return bc::wallet::payment\_address

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::wallet::payment\_address Wallet::childAddress(int a\_index)

{

// Testnet payment address.

return bc::wallet::payment\_address(bc::wallet::ec\_public(childPublicKey(a\_index).point()), 0x6f);

}

/\*\*

\* @brief Returns BIP-32 root key.

\*

\* @return bc::wallet::hd\_private

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::wallet::hd\_private Wallet::showPrivateKey()

{

return m\_masterPrivateKey.encoded();

}

/\*\*

\* @brief Returns child private key at index n of keychain.

\*

\* @param index

\* @return bc::wallet::hd\_private

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::wallet::hd\_private Wallet::showChildPrivateKey(int a\_index)

{

return childPrivateKey(a\_index).encoded();

}

/\*\*

\* @brief Return bitcoin address (Base58 encoded) at index n of keychain.

\*

\* @param a\_index

\* @return bc::wallet::payment\_address

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::wallet::payment\_address

Wallet::getAddress(int a\_index)

{

return childAddress(a\_index).encoded();

}

/\*\*

\* @brief Outputs to console the list of mnemonic code phrases.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

Wallet::showMnemonicCodes()

{

// Validate the mnemonic phrase before sharing it with user.

if(bc::wallet::validate\_mnemonic(m\_mnemonic))

{

std::string mnemonicString = bc::join(m\_mnemonic);

std::cout << mnemonicString << std::endl;

}else{

std::cout << "Mnemonic Invalid!" << std::endl;

}

};

/\*\*

\* @brief Shows relevant keys to the user in console. Used for debugging.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

Wallet::showKeys()

{

showMnemonicCodes();

std::cout << "BIP 32 Root Key: " << showPrivateKey() << std::endl;

std::cout << "Address: " << getAddress(1) << std::endl;

std::cout << "Address: " << getAddress(2) << std::endl;

};

/\*\*

\* @brief Sets the current address index to the last unused address. Prevents address reuse.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

Wallet::set\_address\_index\_to\_last\_unused\_address()

{

while(true)

{

// Check if the given address was used.

if(transactions->calculateBalance(getAddress(m\_address\_index)))

{

m\_address\_index++;

}

else

{

break;

}

}

};

/\*\*

\* @brief Returns balance as unsigned long long.

\*

\* @return unsigned long long represents balance value of wallet.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

unsigned long long

Wallet::getBalance() const

{

return transactions->getBalance();

}

/\*\*

\* @brief Returns balance as string.

\*

\* @return std::string string that represents balance value of wallet.

\* @author Philip Glazman

\* @date 4/28/18

\*/

std::string

Wallet::get\_balance\_as\_string() const

{

return bc::encode\_base10(transactions->getBalance(),8);

};

/\*\*

\* @brief Creates a P2PKH transaction

\*

\* @param a\_address string Address to send value to.

\* @param a\_satoshis unsigned long long Satoshi value to send.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

Wallet::build\_P2PKH(std::string a\_address, unsigned long long a\_satoshis)

{

// Build tx.

bc::wallet::payment\_address address = bc::wallet::payment\_address(a\_address);

bc::chain::transaction tx = transactions->P2PKH(a\_address,a\_satoshis);

// Show tx.

// @TODO - return tx.

transactions->show\_raw\_tx(tx);

// transactions->broadcastTransaction(tx);

};

/\*\*

\* @brief Creates a P2PKH transaction with a given tx fee.

\*

\* @param a\_address string Address to send value to.

\* @param a\_satoshis unsigned long long Satoshi value to send.

\* @param a\_fees

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

Wallet::build\_P2PKH(std::string a\_address, unsigned long long a\_satoshis, unsigned long long a\_fees)

{

// Build tx.

bc::wallet::payment\_address address = bc::wallet::payment\_address(a\_address);

bc::chain::transaction tx = transactions->P2PKH(a\_address,a\_satoshis,a\_fees);

transactions->show\_raw\_tx(tx);

std::cout << tx.inputs()[0].address() << std::endl;

const bc::wallet::payment\_address utxo\_address = transactions->get\_last\_utxo\_address();

bc::data\_chunk public\_key = getPublicKey(utxo\_address);

bc::wallet::hd\_private private\_key = getPrivateKey(utxo\_address);

bc::endorsement signature = transactions->create\_signature(public\_key,private\_key,tx);

bc::chain::script unlocking\_script = transactions->create\_sig\_script(signature,public\_key);

tx.inputs()[0].set\_script(unlocking\_script);

transactions->show\_raw\_tx(tx);

// Broadcast tx.

// transactions->broadcastTransaction(tx);

};

/\*\*

\* @brief

\*

\* @return std::vector< Transaction::m\_tx >

\*

\* @author Philip Glazman

\* date 4/28/18

\*/

std::vector< Transaction::m\_tx >

Wallet::get\_transaction\_history()

{

return transactions->get\_transaction\_history();

}

/\*\*

\* @brief Returns public key with a given payment address.

\*

\* @param a\_address

\* @return bc::data\_chunk

\*

\* @author Philip Glazman

\* @date 4/29/18

\*/

bc::data\_chunk

Wallet::getPublicKey(bc::wallet::payment\_address a\_address)

{

for(int i = 1 ; i < INT\_MAX; i ++ )

{

if(childAddress(i) == a\_address)

{

return bc::to\_chunk(childPublicKey(i).point());

}

}

};

/\*\*

\* @brief Returns private key with a given payment address.

\*

\* @param a\_address

\* @return bc::wallet::hd\_private

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::wallet::hd\_private

Wallet::getPrivateKey(bc::wallet::payment\_address a\_address)

{

for(int i = 1 ; i < INT\_MAX; i ++ )

{

if(childAddress(i) == a\_address)

{

//childPrivateKey

return childPrivateKey(i);

}

}

};

~~~~~~~~~~~~~~~~~~~~~Source code for file transaction.cpp~~~~~~~~~~~~~~~~~~~~~

#include "stdafx.h"

/\*\*

\* @brief Construct a new Transaction:: Transaction object

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

Transaction::Transaction()

{

network = new Network();

unspent\_output = new utxo();

m\_utxoSum = 0;

};

/\*\*

\* @brief Destroy the Transaction:: Transaction object

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

Transaction::~Transaction()

{

delete network;

delete unspent\_output;

}

/\*\*

\* @brief Creates output for a P2PKH transaction.

\*

\* @param a\_address bc::wallet::payment\_address address that owns output

\* @param a\_satoshis unsigned long long value of output

\* @return bc::chain::output

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::chain::output

Transaction::createOutputP2PKH(bc::wallet::payment\_address a\_address, unsigned long long a\_satoshis)

{

// Hash the Public Key of the Address. OP\_DUP OP\_HASH160 <PKH> OP\_EQUALVERIFY OP\_CHECKSIG

bc::chain::script outputScript = bc::chain::script().to\_pay\_key\_hash\_pattern(a\_address.hash());

// to\_pay\_key\_hash\_pattern creates an operation::list. Assignment constructor makes assigns it to outputScript.

bc::chain::output output(a\_satoshis,outputScript);

return output;

};

/\*\*

\* @brief Shows the transaction output.

\*

\* @param output bc::chain::output output point

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

Transaction::showTxOutput(bc::chain::output output)

{

std::cout << "Sending Bitcoin: \nAmount: " << bc::encode\_base10(output.value(), 8) << "BTC : Output Script: " << output.script().to\_string(0) << std::endl;

};

/\*\*

\* @brief Outputs raw transaction into hex.

\*

\* @param a\_transaction

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

Transaction::show\_raw\_tx(bc::chain::transaction a\_transaction)

{

std::cout << "Raw Transaction: " << std::endl;

std::cout << bc::encode\_base16(a\_transaction.to\_data()) << std::endl;

}

/\*\*

\* @brief Creates an approximate size of the transaction in bytes using the number of inputs and outputs.

\*

\* @param inputs, integer. Number of inputs in the transaction.

\* @param outputs, integer. Number of outputs in the transaction.

\* @return int, Number of bytes.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

int

Transaction::calculateTxSize(int inputs, int outputs)

{

// Conservative case, inputs are 181 bytes. Uncompressed public keys vary in size.

// Outputs are 34 bytes.

return inputs\*181+outputs\*34+10;

};

/\*\*

\* @brief Calculates transaction fee for a given transaction size.

\*

\* @param estimated\_tx\_size int size of the transaction in bytes.

\* @return unsigned long long

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

unsigned long long

Transaction::calculate\_tx\_fee(int estimated\_tx\_size)

{

// Refresh fee recommendations.

network->refreshFeeRecommendations();

// Satoshis/Bytes \* Bytes

unsigned long long fees = (unsigned long long)(estimated\_tx\_size \* network->getHourFee());

return fees;

};

/\*\*

\* @brief Create a Meta Data Tx object

\*

\* @return true

\* @return false

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bool

createMetaDataTx()

{

// OP Return tx

std::string messageString = "helloworld";

bc::data\_chunk data(80);

auto source = bc::make\_safe\_deserializer(data.begin(),data.end());

auto sink = bc::make\_unsafe\_serializer(data.begin());

sink.write\_string(messageString);

const auto nullData = source.read\_bytes(80);

std::cout << "Message: " << std::endl;

std::cout << bc::encode\_base16(nullData) << std::endl;

bc::chain::output output2 = bc::chain::output();

output2.set\_script(bc::chain::script(bc::chain::script().to\_null\_data\_pattern(nullData)));

output2.set\_value(0);

return true;

};

/\*\*

\* @brief Create a Signature object

\*

\* @param a\_pubKey

\* @param a\_privKey

\* @param a\_transaction

\* @return bc::endorsement

\*

\* @author Philip Glazman

\* @date 4/29/18

\*/

bc::endorsement

Transaction::create\_signature(bc::data\_chunk a\_pubKey, bc::ec\_secret a\_privKey,bc::chain::transaction a\_transaction)

{

bc::chain::script lockingScript = bc::chain::script().to\_pay\_key\_hash\_pattern(bc::bitcoin\_short\_hash(a\_pubKey));

bc::endorsement signature;

if(lockingScript.create\_endorsement(signature, a\_privKey, lockingScript, a\_transaction, 0u, bc::machine::all))

{

std::cout << "Signature: " << std::endl;

std::cout << bc::encode\_base16(signature) << "\n" << std::endl;

return signature;

}

else

{

Error::RecordError(std::string("Cannot create signature endorsement."));

Error::DisplayErrors();

}

};

/\*\*

\* @brief Create a sig script object

\*

\* @param a\_signature

\* @param a\_pubKey

\* @return bc::script

\*

\* @author Philip Glazman

\* @date 4/29/18

\*/

bc::chain::script

Transaction::create\_sig\_script(bc::endorsement a\_signature, bc::data\_chunk a\_pubKey)

{

bc::machine::operation::list signature\_script;

signature\_script.push\_back(bc::machine::operation(a\_signature));

signature\_script.push\_back(bc::machine::operation(a\_pubKey));

bc::chain::script unlocking\_script(signature\_script);

return unlocking\_script;

}

/\*\*

\* @brief Constructs P2PKH script transaction.

\*

\* @param a\_publicKey, public key address of bitcoin payment address to use.

\* @param a\_privKey

\* @param a\_destinationAddress

\* @param a\_satoshis

\* @return true

\* @return false

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::chain::transaction

Transaction::P2PKH(bc::wallet::payment\_address a\_destinationAddress, unsigned long long a\_satoshis)

{

unsigned long long input\_value = 0;

unsigned long long change\_value = 0;

// Start building Transaction.

// Instantiate the transaction object.

bc::chain::transaction tx = bc::chain::transaction();

// Set transaction version.

// uint32\_t version = 1u;

// tx.set\_version(version);

// Find unspent output.

m\_utxo utxo\_to\_spend = unspent\_output -> find\_utxo(a\_satoshis);

// Create inputs.

// For each input, point to unspent transaction output.

for( const auto &utxo : utxo\_to\_spend)

{

// Create input.

bc::chain::input input = bc::chain::input();

bc::hash\_digest utxo\_hash = std::get<1>(utxo);

bc::chain::output\_point previous\_output(utxo\_hash,0);

input.set\_previous\_output(previous\_output);

input.set\_sequence(0xffffffff);

tx.inputs().push\_back(input);

input\_value += std::get<0>(utxo);

// Get pub\_key.

m\_last\_utxo\_address = std::get<2>(utxo);

};

change\_value = input\_value - a\_satoshis;

// Find recommended fee.

int estimated\_tx\_bytes = calculateTxSize(tx.inputs().size(), 2);

unsigned long long fees = calculate\_tx\_fee(estimated\_tx\_bytes);

// Subtract fees from the change.

if( change\_value - fees > 0)

{

change\_value -= fees;

std::cout << "fees to send: " <<fees << "change value" << change\_value <<std::endl;

bc::wallet::payment\_address change\_address= std::get<2>(utxo\_to\_spend[0]);

tx.outputs().push\_back(createOutputP2PKH(change\_address,change\_value));

}

// If fees are greater than change, make change 0.

else if (change\_value - fees <= 0)

{

change\_value = 0;

}

// Create output.

tx.outputs().push\_back(createOutputP2PKH(a\_destinationAddress,a\_satoshis));

// Sign Transaction

// bc::endorsement signature = create\_signature(pub\_key,privKey,tx)

// bc::script = create\_sig\_script(signature,pub\_key)

// tx.inputs()[0].set\_script(unlocking\_script)

// Return transaction.

return tx;

};

/\*\*

\* @brief Constructs P2PKH script transaction.

\*

\* @param a\_publicKey, public key address of bitcoin payment address to use.

\* @param a\_privKey

\* @param a\_destinationAddress

\* @param a\_satoshis

\* @param a\_fees

\* @return true

\* @return false

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::chain::transaction

Transaction::P2PKH(bc::wallet::payment\_address a\_destinationAddress, unsigned long long a\_satoshis, unsigned long long a\_fees)

{

unsigned long long input\_value = 0;

unsigned long long change\_value = 0;

// Start building Transaction.

// Instantiate the transaction object.

bc::chain::transaction tx = bc::chain::transaction();

// Set transaction version.

// uint32\_t version = 1u;

// tx.set\_version(version);

// Find unspent output.

m\_utxo utxo\_to\_spend = unspent\_output -> find\_utxo(a\_satoshis);

// Create inputs.

// For each input, point to unspent transaction output.

for( const auto &utxo : utxo\_to\_spend)

{

// Create input.

bc::chain::input input = bc::chain::input();

bc::hash\_digest utxo\_hash = std::get<1>(utxo);

bc::chain::output\_point previous\_output(utxo\_hash,0);

input.set\_previous\_output(previous\_output);

input.set\_sequence(0xffffffff);

tx.inputs().push\_back(input);

input\_value += std::get<0>(utxo);

m\_last\_utxo\_address = std::get<2>(utxo);

};

change\_value = input\_value - a\_satoshis;

// Find recommended fee.

int estimated\_tx\_bytes = calculateTxSize(tx.inputs().size(), 2);

unsigned long long fees = estimated\_tx\_bytes \* a\_fees;

// Subtract fees from the change.

if( change\_value - fees > 0)

{

change\_value -= fees;

std::cout << "fees to send: " <<fees << "change value" << change\_value <<std::endl;

bc::wallet::payment\_address change\_address= std::get<2>(utxo\_to\_spend[0]);

tx.outputs().push\_back(createOutputP2PKH(change\_address,change\_value));

}

// If fees are greater than change, make change 0.

else if (change\_value - fees <= 0)

{

change\_value = 0;

}

// Create output.

tx.outputs().push\_back(createOutputP2PKH(a\_destinationAddress,a\_satoshis));

// Return transaction.

return tx;

};

/\*\*

\* @brief Returns balanace of n payment address.

\*

\* @param a\_address, payment address.

\* @return unsigned long long

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

unsigned long long

Transaction::getBalanceForAddress(bc::wallet::payment\_address a\_address)

{

unsigned long long utxo = 0;

bc::hash\_digest utxo\_hash;

// Connect to libbitcoin servers.

bc::client::obelisk\_client &rpc = network->connect();

// Lambda callback function for getting utxo for addy.

static const auto on\_done = [this, &utxo,&a\_address,&utxo\_hash](const bc::chain::history::list& rows)

{

// For each row in chain history, check for balance.

for(const auto& row: rows)

{

// Unspent transaction output.

if (row.spend.hash() == bc::null\_hash)

{

utxo += row.value;

utxo\_hash = row.output.hash();

std::cout << bc::encode\_hash(utxo\_hash) << std::endl;

unspent\_output -> add\_transaction(row.value, utxo\_hash, a\_address);

}

// Spent transaction output.

{

if(row.spend.hash() != bc::null\_hash)

{

m\_transactions.push\_back(std::make\_tuple(row.value,row.spend.hash(),row.spend\_height));

}

if(row.output.hash() != bc::null\_hash)

{

m\_transactions.push\_back(std::make\_tuple(row.value,row.output.hash(),row.output\_height));

}

}

}

};

static const auto on\_error = [](const bc::code ec)

{

Error::RecordError(std::string("Error connecting to bitcoin network."));

};

// Get Blockchain history on this address.

rpc.blockchain\_fetch\_history3(on\_error, on\_done, a\_address);

// Wait for history to be fetched.

rpc.wait();

network -> disconnect();

m\_utxoMap[a\_address] = std::make\_pair(utxo\_hash, utxo);

// Return utxo for a\_address.

return utxo;

};

/\*\*

\* @brief

\*

\* @param a\_address, payment address to check UTXO for.

\* @param a\_amount, minimum value of satoshis needed in UTXO.

\* @return bc::chain::points\_value

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bc::chain::points\_value

Transaction::getUTXOs(bc::wallet::payment\_address a\_address, unsigned long long a\_amount)

{

// Connect to libbitcoin servers.

bc::client::obelisk\_client &rpc = network->connect();

bc::chain::points\_value val1;

static const auto on\_done = [&val1](const bc::chain::points\_value& vals) {

std::cout << "Success: " << vals.value() << std::endl;

val1 = vals;

};

static const auto on\_error = [](const bc::code& ec) {

std::cout << "Error Code: " << ec.message() << std::endl;

};

rpc.blockchain\_fetch\_unspent\_outputs(on\_error, on\_done, a\_address, a\_amount, bc::wallet::select\_outputs::algorithm::greedy);

rpc.wait();

network -> disconnect();

//return allPoints;

return val1;

};

/\*\*

\* @brief Broadcasts transaction to the network.

\*

\* @param tx

\* @return true

\* @return false

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bool

Transaction::broadcastTransaction(bc::chain::transaction tx)

{

// Connect to libbitcoin servers.

bc::client::obelisk\_client &rpc = network->connect();

static const auto on\_done = [](const bc::code& ec) {

std::cout << "Success: " << ec.message() << std::endl;

};

static const auto on\_error = [](const bc::code& ec) {

std::cout << "Error Code: " << ec.message() << std::endl;

};

rpc.transaction\_pool\_broadcast(on\_error, on\_done, tx);

rpc.wait();

network -> disconnect();

return true;

};

/\*\*

\* @brief Checks if the given payment address has recieved any bitcoin in its history.

\*

\* @param a\_address, address to check.

\* @return true, payment address has recieved bitcoin.

\* @return false, payment address has never recieved bitcoin.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bool

Transaction::isAddressUsed(bc::wallet::payment\_address a\_address)

{

// Satoshis recieved.

unsigned long long recieved = 0;

// Connect to libbitcoin servers.

bc::client::obelisk\_client &rpc = network->connect();

// Lambda callback function for getting utxo for addy.

static const auto on\_done = [&recieved](const bc::chain::history::list& rows)

{

// For each row in chain history, check for balance.

for(const auto& row: rows)

{

recieved += row.value;

}

};

static const auto on\_error = [](const bc::code ec)

{

Error::RecordError(std::string("Error connecting to bitcoin network."));

};

// Get Blockchain history on this address.

rpc.blockchain\_fetch\_history3(on\_error, on\_done, a\_address);

// Wait for history to be fetched.

rpc.wait();

network -> disconnect();

// If address recieved any bitcoin, than it is used.

if(recieved > 0)

{

return true;

}

else

{

return false;

}

};

/\*\*

\* @brief Calculates the balance of the wallet. Atlas calls this function until false is returned.

\*

\* @param a\_address, payment address to check if address is used, and add any existing balance.

\* @return true, address is used.

\* @return false, address is not used.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

bool

Transaction::calculateBalance(bc::wallet::payment\_address a\_address)

{

std::cout << "Checking balance for " << a\_address << std::endl;

// Check if address is used.

if(isAddressUsed(a\_address))

{

// Get balance for the address. Add it to the sum.

m\_utxoSum += getBalanceForAddress(a\_address);

return true;

}

else

{

show\_transaction\_history();

return false;

}

};

/\*\*

\* @brief Selector for current balance of utxo.

\*

\* @return unsigned long long

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

unsigned long long

Transaction::getBalance() const

{

return m\_utxoSum;

};

/\*\*

\* @brief Get the transaction history.

\*

\* @return std::vector< m\_tx >

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

std::vector< Transaction::m\_tx >

Transaction::get\_transaction\_history() const

{

return m\_transactions;

};

/\*\*

\* @brief Shows each transaction in transaction history.

\*

\* @author Philip Glazman

\* @date 4/28/18

\*/

void

Transaction::show\_transaction\_history()

{

std::sort(m\_transactions.begin(),m\_transactions.end(),compare\_block\_height);

for(int i = 0; i < m\_transactions.size(); i++)

{

std::cout << std::get<0>(m\_transactions[i]) << bc::encode\_hash(std::get<1>(m\_transactions[i])) << std::get<2>(m\_transactions[i]) << std::endl;

}

};

/\*\*

\* @brief Comparator function for comparing block height between two transactions. Used for sorting.

\*

\* @param a

\* @param b

\* @return true

\* @return false

\*

\* @author Philip Glazman

\* @date 4/28/2018

\*/

bool

Transaction::compare\_block\_height(const m\_tx &a, const m\_tx &b)

{

return std::get<2>(a) > std::get<2>(b);

};

/\*\*

\* @brief Get the last utxo address object

\*

\* @return bc::wallet::payment\_address

\*

\* @author Philip Glazman

\* @date 4/29/18

\*/

bc::wallet::payment\_address

Transaction::get\_last\_utxo\_address() const

{

return m\_last\_utxo\_address;

}

~~~~~~~~~~~~~~~~~~~~~Source code for file valid\_address.cpp~~~~~~~~~~~~~~~~~~~~~

#include "valid\_address.hpp"

#include <string>

#include "/usr/local/include/openssl/sha.h"

bool valid\_address::valid(std::string a\_address)

{

unsigned char dec[32], d1[SHA256\_DIGEST\_LENGTH], d2[SHA256\_DIGEST\_LENGTH];

return true;

}

~~~~~~~~~~~~~~~~~~~~~Source code for file debug.cpp~~~~~~~~~~~~~~~~~~~~~

#include "valid\_address.hpp"

#include <iostream>

int main()

{

std::cout << valid\_address::valid("hello") << std::endl;

}

# Test Cases

Launch new wallet.

Restore existing wallet using mnemonic phrase:

|  |  |
| --- | --- |
| scatter | blanket |
| found | mother |
| issue | frequent |
| friend | acid |
| front | shaft |
| glare | loud |

1. Sending a transaction to the address located on the main tab.

2. Check if the transaction was successfully broadcasted using <https://live.blockcypher.com/btc-testnet/>

3. Test a simple bitcoin script:

Witness: 1 2

Witness Script: OP\_ADD 3 OP\_EQUAL

4. Compare the history of transactions in the history tab with the blockcypher explorer.